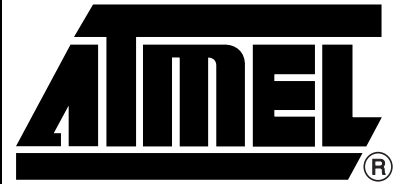


Features

- 80C51 Core Architecture
- 256 Bytes of On-chip RAM
- 256 Bytes of On-chip XRAM
- 16K Bytes of On-chip Flash Memory
 - Data Retention: 10 Years at 85°C
 - Erase/Write Cycle: 100K
- Boot Code Section with Independent Lock Bits
- 2K Bytes of On-chip Flash for Bootloader
- In-System Programming by On-Chip Boot Program (CAN, UART) and IAP Capability
- 2K Bytes of On-chip EEPROM
 - Erase/Write Cycle: 100K
- 14-sources 4-level Interrupts
- Three 16-bit Timers/Counters
- Full Duplex UART Compatible 80C51
- Maximum Crystal Frequency 40 MHz. In X2 Mode, 20 MHz (CPU Core, 40 MHz)
- Three or Four Ports: 16 or 20 Digital I/O Lines
- Two-channel 16-bit PCA
 - PWM (8-bit)
 - High-speed Output
 - Timer and Edge Capture
- Double Data Pointer
- 21-bit Watchdog Timer (7 Programmable bits)
- A 10-bit Resolution Analog-to-Digital Converter (ADC) with 8 Multiplexed Inputs
- Full CAN Controller
 - Fully Compliant with CAN rev.# 2.0A and 2.0B
 - Optimized Structure for Communication Management (Via SFR)
 - 4 Independent Message Objects
 - Each Message Object Programmable on Transmission or Reception
 - Individual Tag and Mask Filters up to 29-bit Identifier/Channel
 - 8-byte Cyclic Data Register (FIFO)/Message Object
 - 16-bit Status and Control Register/Message Object
 - 16-bit Time-Stamping Register/Message Object
 - CAN Specification 2.0 Part A or 2.0 Part B Programmable for Each Message Object
 - Access to Message Object Control and Data Registers Via SFR
 - Programmable Reception Buffer Length up to 4 Message Objects
 - Priority Management of Reception of Hits on Several Message Objects Simultaneously (Basic CAN Feature)
 - Priority Management for Transmission
 - Message Object Overrun Interrupt
 - Supports
 - Time Triggered Communication
 - Autobaud and Listening Mode
 - Programmable Automatic Reply Mode
- 1-Mbit/s Maximum Transfer Rate at 8 MHz⁽¹⁾ Crystal Frequency In X2 Mode
- Readable Error Counters
- Programmable Link to On-chip Timer for Time Stamping and Network Synchronization
- Independent Baud Rate Prescaler
- Data, Remote, Error and Overload Frame Handling
- Power-saving Modes
 - Idle Mode
 - Power-down Mode
- Power Supply: 3 Volts to 5.5 Volts
- Temperature Range: Industrial (-40° to +85°C)
- Packages: SOIC28, SOIC24, PLCC28, VQFP32

Note: 1. At BRP = 1 sampling point will be fixed.



Enhanced 8-bit Microcontroller with CAN Controller and Flash

T89C51CC02
AT89C51CC02

Rev. 4126L-CAN-01/08



Description

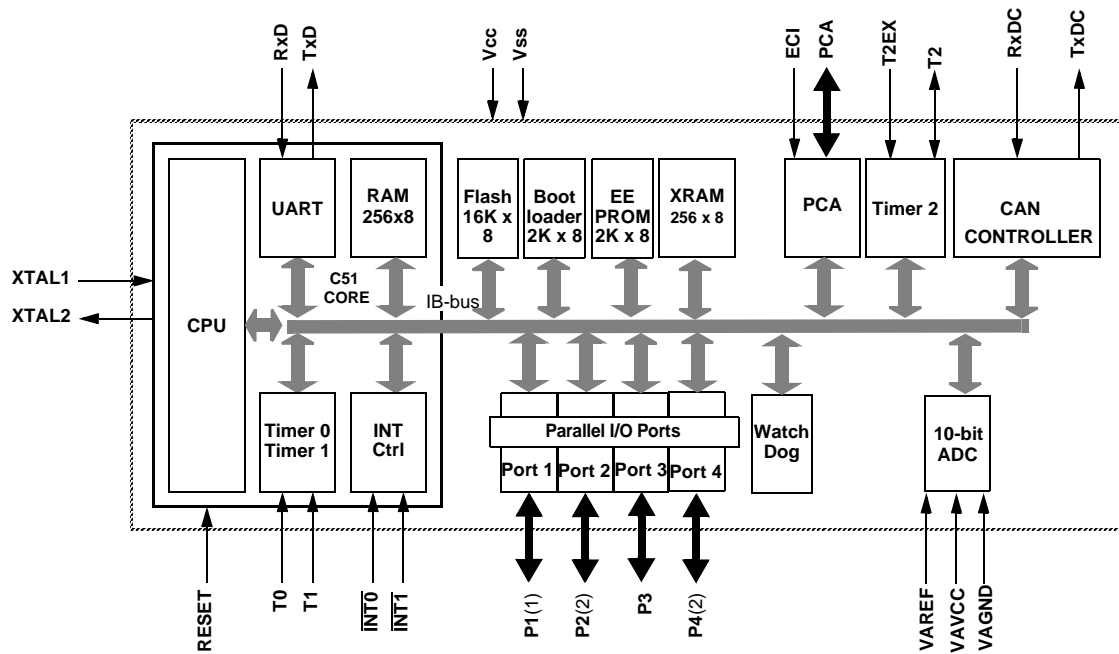
Part of the CANary™ family of 8-bit microcontrollers dedicated to CAN network applications, the T89C51CC02 is a low-pin count 8-bit Flash microcontroller.

In X2 Mode a maximum external clock rate of 20 MHz reaches a 300 ns cycle time.

Besides the full CAN controller T89C51CC02 provides 16K Bytes of Flash memory including In-System Programming (ISP), 2K Bytes Boot Flash Memory, 2K Bytes EEPROM and 512 Bytes RAM.

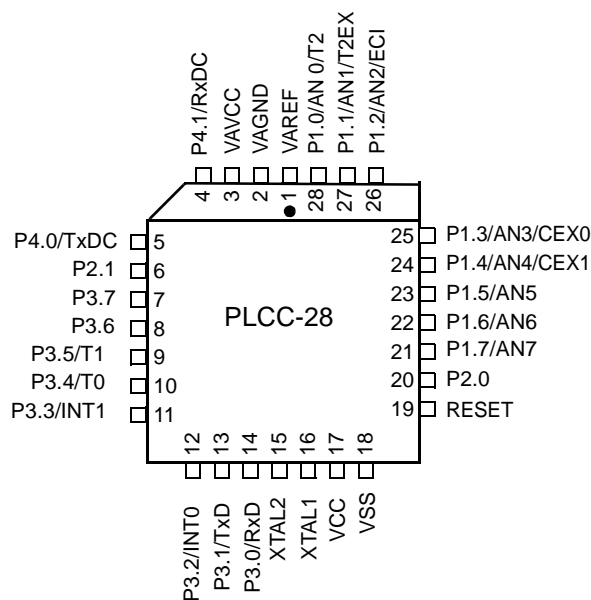
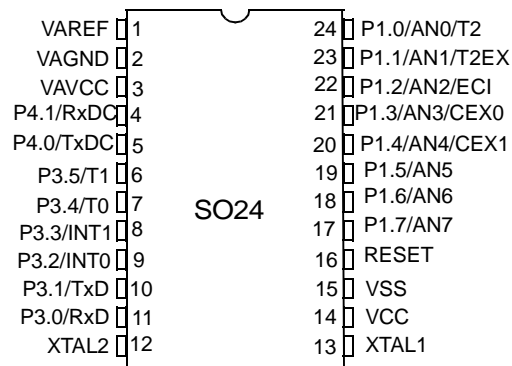
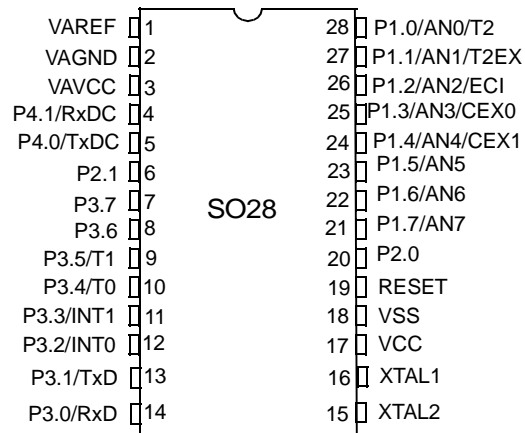
Special attention is payed to the reduction of the electro-magnetic emission of T89C51CC02.

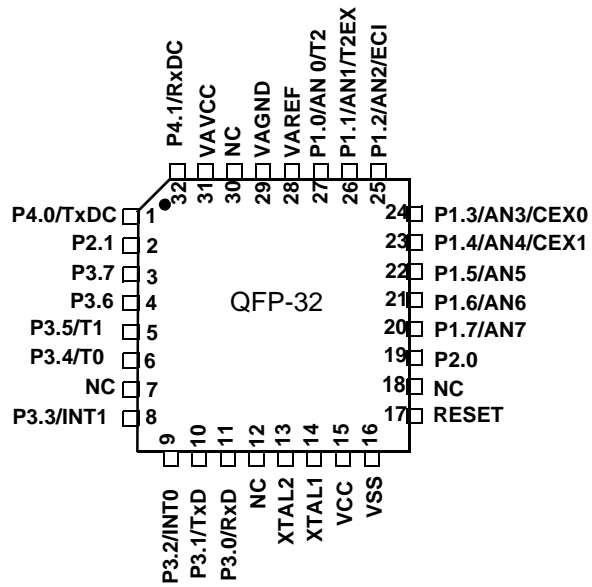
Block Diagram



Note: 1. 8 analog Inputs/8 Digital I/O.
2. 2-bit I/O Port.

Pin Configurations





Pin Description

| Pin Name | Type | Description |
|----------|------|--|
| VSS | GND | Circuit ground |
| VCC | | Supply Voltage |
| VAREF | | Reference Voltage for ADC (input) |
| VAVCC | | Supply Voltage for ADC |
| VAGND | | Reference Ground for ADC (internally connected with the VSS) |
| P1.0:7 | I/O | <p>Port 1:</p> <p>Is an 8-bit bi-directional I/O port with internal pull-ups. Port 1 pins can be used for digital input/output or as analog inputs for the Analog Digital Converter (ADC). Port 1 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 1 pins that are being pulled low externally will be the source of current (I_{IL}, See section 'Electrical Characteristic') because of the internal pull-ups. Port 1 pins are assigned to be used as analog inputs via the ADCCF register (in this case the internal pull-ups are disconnected).</p> <p>As a secondary digital function, port 1 contains the Timer 2 external trigger and clock input; the PCA external clock input and the PCA module I/O.</p> <p>P1.0/AN0/T2 Analog input channel 0, External clock input for Timer/counter2.</p> <p>P1.1/AN1/T2EX Analog input channel 1, Trigger input for Timer/counter2.</p> <p>P1.2/AN2/ECI Analog input channel 2, PCA external clock input.</p> <p>P1.3/AN3/CEX0 Analog input channel 3, PCA module 0 Entry of input/PWM output.</p> <p>P1.4/AN4/CEX1 Analog input channel 4, PCA module 1 Entry of input/PWM output.</p> <p>P1.5/AN5 Analog input channel 5,</p> <p>P1.6/AN6 Analog input channel 6,</p> <p>P1.7/AN7 Analog input channel 7, It can drive CMOS inputs without external pull-ups.</p> |
| P2.0:1 | I/O | <p>Port 2:</p> <p>Is an 2-bit bi-directional I/O port with internal pull-ups. Port 2 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 2 pins that are being pulled low externally will be a source of current (I_{IL}, on the datasheet) because of the internal pull-ups. In the T89C51CC02 Port 2 can sink or source 5mA. It can drive CMOS inputs without external pull-ups.</p> |

| Pin Name | Type | Description |
|----------|------|---|
| P3.0:7 | I/O | <p>Port 3: Is an 8-bit bi-directional I/O port with internal pull-ups. Port 3 pins that have 1's written to them are pulled high by the internal pull-up transistors and can be used as inputs in this state. As inputs, Port 3 pins that are being pulled low externally will be a source of current (I_{IL}, See section 'Electrical Characteristic') because of the internal pull-ups.</p> <p>The output latch corresponding to a secondary function must be programmed to one for that function to operate (except for TxD and \overline{WR}). The secondary functions are assigned to the pins of port 3 as follows: P3.0/RxD: Receiver data input (asynchronous) or data input/output (synchronous) of the serial interface P3.1/TxD: Transmitter data output (asynchronous) or clock output (synchronous) of the serial interface P3.2/$\overline{INT0}$: External interrupt 0 input/timer 0 gate control input P3.3/$\overline{INT1}$: External interrupt 1 input/timer 1 gate control input P3.4/T0: Timer 0 counter input P3.5/T1: Timer 1 counter input P3.6: Regular I/O port pin P3.7: Regular I/O port pin</p> |
| P4.0:1 | I/O | <p>Port 4: Is an 2-bit bi-directional I/O port with internal pull-ups. Port 4 pins that have 1's written to them are pulled high by the internal pull-ups and can be used as inputs in this state. As inputs, Port 4 pins that are being pulled low externally will be a source of current (I_{IL}, on the datasheet) because of the internal pull-up transistor.</p> <p>The output latch corresponding to a secondary function RxDC must be programmed to one for that function to operate. The secondary functions are assigned to the two pins of port 4 as follows: P4.0/TxDC: Transmitter output of CAN controller P4.1/RxDC: Receiver input of CAN controller. It can drive CMOS inputs without external pull-ups.</p> |
| RESET | I/O | <p>Reset: A high level on this pin during two machine cycles while the oscillator is running resets the device. An internal pull-down resistor to VSS permits power-on reset using only an external capacitor to VCC.</p> |
| XTAL1 | I | <p>XTAL1: Input of the inverting oscillator amplifier and input of the internal clock generator circuits. To drive the device from an external clock source, XTAL1 should be driven, while XTAL2 is left unconnected. To operate above a frequency of 16 MHz, a duty cycle of 50% should be maintained.</p> |
| XTAL2 | O | <p>XTAL2: Output from the inverting oscillator amplifier.</p> |

I/O Configurations

Each Port SFR operates via type-D latches, as illustrated in Figure 1 for Ports 3 and 4. A CPU 'write to latch' signal initiates transfer of internal bus data into the type-D latch. A CPU 'read latch' signal transfers the latched Q output onto the internal bus. Similarly, a 'read pin' signal transfers the logical level of the Port pin. Some Port data instructions activate the 'read latch' signal while others activate the 'read pin' signal. Latch instructions are referred to as Read-Modify-Write instructions. Each I/O line may be independently programmed as input or output.

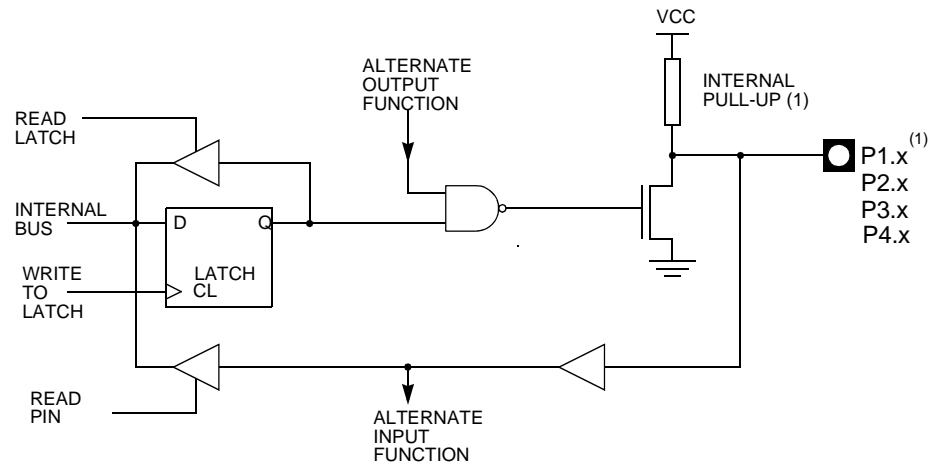
Port Structure

Figure 1 shows the structure of Ports, which have internal pull-ups. An external source can pull the pin low. Each Port pin can be configured either for general-purpose I/O or for its alternate input output function.

To use a pin for general-purpose output, set or clear the corresponding bit in the Px register (x = 1 to 4). To use a pin for general-purpose input, set the bit in the Px register. This turns off the output FET drive.

To configure a pin for its alternate function, set the bit in the Px register. When the latch is set, the 'alternate output function' signal controls the output level (See Figure 1). The operation of Ports is discussed further in 'Quasi-Bi-directional Port Operation' paragraph.

Figure 1. Ports Structure



Note: 1. The internal pull-up can be disabled on P1 when analog function is selected.

Read-Modify-Write Instructions

Some instructions read the latch data rather than the pin data. The latch based instructions read the data, modify the data and then rewrite the latch. These are called 'Read-Modify-Write' instructions. Below is a complete list of these special instructions (See Table 1). When the destination operand is a Port or a Port bit, these instructions read the latch rather than the pin:

Table 1. Read/Modify/Write Instructions

| Instruction | Description | Example |
|-------------|-----------------------------------|-----------------|
| ANL | Logical AND | ANL P1, A |
| ORL | Logical OR | ORL P2, A |
| XRL | Logical EX-OR | XRL P3, A |
| JBC | Jump if bit = 1 and clear bit | JBC P1.1, LABEL |
| CPL | Complement bit | CPL P3.0 |
| INC | Increment | INC P2 |
| DEC | Decrement | DEC P2 |
| DJNZ | Decrement and jump if not zero | DJNZ P3, LABEL |
| MOV Px.y, C | Move carry bit to bit y of Port x | MOV P1.5, C |
| CLR Px.y | Clear bit y of Port x | CLR P2.4 |
| SET Px.y | Set bit y of Port x | SET P3.3 |

It is not obvious that the last three instructions in this list are Read-Modify-Write instructions. These instructions read the port (all 8 bits), modify the specifically addressed bit and write the new byte back to the latch. These Read-Modify-Write instructions are directed to the latch rather than the pin in order to avoid possible misinterpretation of voltage (and therefore, logic) levels at the pin. For example, a Port bit used to drive the base of an external bipolar transistor cannot rise above the transistor's base-emitter junction voltage (a value lower than VIL). With a logic one written to the bit, attempts by the CPU to read the Port at the pin are misinterpreted as logic zero. A read of the latch rather than the pins returns the correct logic one value.

Quasi Bi-directional Port Operation

Port 1, Port 3 and Port 4 have fixed internal pull-ups and are referred to as 'quasi-bi-directional' Ports. When configured as an input, the pin impedance appears as logic one and sources current in response to an external logic zero condition. Resets write logic one to all Port latches. If logical zero is subsequently written to a Port latch, it can be returned to input conditions by a logic one written to the latch.

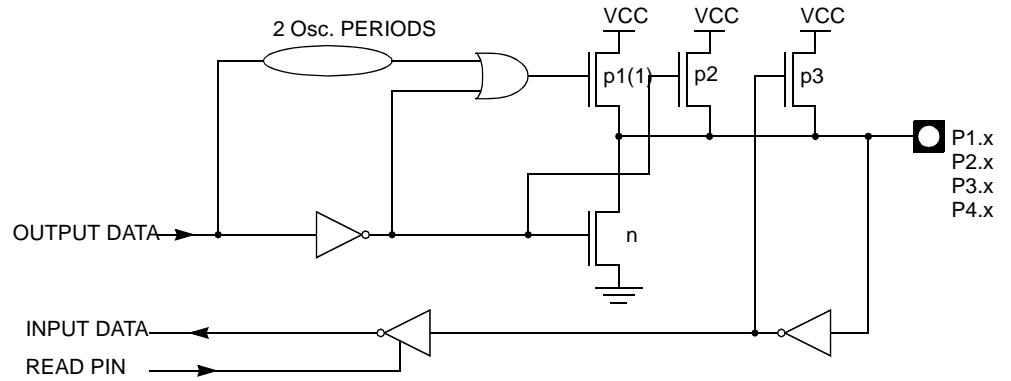
Note: Port latch values change near the end of Read-Modify-Write instruction cycles. Output buffers (and therefore the pin state) are updated early in the instruction after Read-Modify-Write instruction cycle.

Logical zero-to-one transitions in Port 1, Port 3 and Port 4 use an additional pull-up (p1) to aid this logic transition See Figure 2. This increases switch speed. This extra pull-up sources 100 times normal internal circuit current during 2 oscillator clock periods. The internal pull-ups are field-effect transistors rather than linear resistors. Pull-ups consist of three p-channel FET (pFET) devices. A pFET is on when the gate senses logic zero and off when the gate senses logic one. pFET #1 is turned on for two oscillator periods immediately after a zero-to-one transition in the Port latch. A logic one at the Port pin turns on pFET #3 (a weak pull-up) through the inverter. This inverter and pFET pair form a latch to drive logic one. pFET #2 is a very weak pull-up switched on whenever the

associated nFET is switched off. This is traditional CMOS switch convention. Current strengths are 1/10 that of pFET #3.

Note: During Reset, pFET#1 is not activated. During Reset, only the weak pFET#3 pull up the pin.

Figure 2. Internal Pull-up Configurations



SFR Mapping

Tables 3 through Table 11 show the Special Function Registers (SFRs) of the T89C51CC02.

Table 2. C51 Core SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|---------------------------------------|----|----|----|-----|-----|----|----|---|
| ACC | E0h | Accumulator | | | | | | | | |
| B | F0h | B Register | | | | | | | | |
| PSW | D0h | Program Status Word | CY | AC | F0 | RS1 | RS0 | OV | F1 | P |
| SP | 81h | Stack Pointer | | | | | | | | |
| DPL | 82h | Data Pointer Low byte LSB of DPTR | | | | | | | | |
| DPH | 83h | Data Pointer High byte MSB of DPTR | | | | | | | | |

Table 3. I/O Port SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|-------------|---|---|---|---|---|---|---|---|
| P1 | 90h | Port 1 | | | | | | | | |
| P2 | A0h | Port 2 (x2) | | | | | | | | |
| P3 | B0h | Port 3 | | | | | | | | |
| P4 | C0h | Port 4 (x2) | | | | | | | | |

Table 4. Timers SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|-------------------------------|-------|-------|-----|-----|-------|-------|-----|-----|
| TH0 | 8Ch | Timer/Counter 0 High byte | | | | | | | | |
| TL0 | 8Ah | Timer/Counter 0 Low byte | | | | | | | | |
| TH1 | 8Dh | Timer/Counter 1 High byte | | | | | | | | |
| TL1 | 8Bh | Timer/Counter 1 Low byte | | | | | | | | |
| TH2 | CDh | Timer/Counter 2 High byte | | | | | | | | |
| TL2 | CCh | Timer/Counter 2 Low byte | | | | | | | | |
| TCON | 88h | Timer/Counter 0 and 1 control | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| TMOD | 89h | Timer/Counter 0 and 1 Modes | GATE1 | C/T1# | M11 | M01 | GATE0 | C/T0# | M10 | M00 |

Table 4. Timers SFRs (Continued)

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--|-----|------|------|------|-------|-----|-------|---------|
| T2CON | C8h | Timer/Counter 2 control | TF2 | EXF2 | RCLK | TCLK | EXEN2 | TR2 | C/T2# | CP/RL2# |
| T2MOD | C9h | Timer/Counter 2 Mode | | | | | | | T2OE | DCEN |
| RCAP2H | CBh | Timer/Counter 2 Reload/Capture High byte | | | | | | | | |
| RCAP2L | CAh | Timer/Counter 2 Reload/Capture Low byte | | | | | | | | |
| WDTRST | A6h | WatchDog Timer Reset | | | | | | | | |
| WDTPRG | A7h | WatchDog Timer Program | | | | | | S2 | S1 | S0 |

Table 5. Serial I/O Port SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--------------------|--------|-----|-----|-----|-----|-----|----|----|
| SCON | 98h | Serial Control | FE/SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
| SBUF | 99h | Serial Data Buffer | | | | | | | | |
| SADEN | B9h | Slave Address Mask | | | | | | | | |
| SADDR | A9h | Slave Address | | | | | | | | |

Table 6. PCA SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--------------------------------|---------|---------|---------|---------|---------|---------|---------|---------|
| CCON | D8h | PCA Timer/Counter Control | CF | CR | | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 |
| CMOD | D9h | PCA Timer/Counter Mode | CIDL | | | | | CPS1 | CPS0 | ECF |
| CL | E9h | PCA Timer/Counter Low byte | | | | | | | | |
| CH | F9h | PCA Timer/Counter High byte | | | | | | | | |
| CCAPM0 | DAh | PCA Timer/Counter Mode 0 | | ECOM0 | CAPP0 | CAPN0 | MAT0 | TOG0 | PWM0 | ECCF0 |
| CCAPM1 | DBh | PCA Timer/Counter Mode 1 | | ECOM1 | CAPP1 | CAPN1 | MAT1 | TOG1 | PWM1 | ECCF1 |
| CCAP0H | FAh | PCA Compare Capture Module 0 H | CCAP0H7 | CCAP0H6 | CCAP0H5 | CCAP0H4 | CCAP0H3 | CCAP0H2 | CCAP0H1 | CCAP0H0 |
| CCAP1H | FBh | PCA Compare Capture Module 1 H | CCAP1H7 | CCAP1H6 | CCAP1H5 | CCAP1H4 | CCAP1H3 | CCAP1H2 | CCAP1H1 | CCAP1H0 |

Table 6. PCA SFRs (Continued)

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--------------------------------|---------|---------|---------|---------|---------|---------|---------|---------|
| CCAP0L | EAh | PCA Compare Capture Module 0 L | CCAP0L7 | CCAP0L6 | CCAP0L5 | CCAP0L4 | CCAP0L3 | CCAP0L2 | CCAP0L1 | CCAP0L0 |
| CCAP1L | EBh | PCA Compare Capture Module 1 L | CCAP1L7 | CCAP1L6 | CCAP1L5 | CCAP1L4 | CCAP1L3 | CCAP1L2 | CCAP1L1 | CCAP1L0 |

Table 7. Interrupt SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|-----------------------------------|----|------|------|-----|------|-------|-------|-------|
| IEN0 | A8h | Interrupt Enable Control 0 | EA | EC | ET2 | ES | ET1 | EX1 | ET0 | EX0 |
| IEN1 | E8h | Interrupt Enable Control 1 | | | | | | ETIM | EADC | ECAN |
| IPL0 | B8h | Interrupt Priority Control Low 0 | | PPC | PT2 | PS | PT1 | PX1 | PT0 | PX0 |
| IPH0 | B7h | Interrupt Priority Control High 0 | | PPCH | PT2H | PSH | PT1H | PX1H | PT0H | PX0H |
| IPL1 | F8h | Interrupt Priority Control Low 1 | | | | | | POVRL | PADCL | PCANL |
| IPH1 | F7h | Interrupt Priority Control High 1 | | | | | | POVRH | PADCH | PCANH |

Table 8. ADC SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--------------------|-------|--------|-------|-------|-------|-------|-------|-------|
| ADCON | F3h | ADC Control | | PSIDLE | ADEN | ADEOC | ADSST | SCH2 | SCH1 | SCH0 |
| ADCF | F6h | ADC Configuration | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| ADCLK | F2h | ADC Clock | | | | PRS4 | PRS3 | PRS2 | PRS1 | PRS0 |
| ADDH | F5h | ADC Data High byte | ADAT9 | ADAT8 | ADAT7 | ADAT6 | ADAT5 | ADAT4 | ADAT3 | ADAT2 |
| ADDL | F4h | ADC Data Low byte | | | | | | | ADAT1 | ADAT0 |

Table 9. CAN SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|-----------------------|-------|-------|--------|---------|----------|-------|-------|------|
| CANGCON | ABh | CAN General Control | ABRQ | OVRQ | TTC | SYNCTTC | AUT-BAUD | TEST | ENA | GRES |
| CANGSTA | AAh | CAN General Status | | OVFG | | TBSY | RBSY | ENFG | BOFF | ERRP |
| CANGIT | 9Bh | CAN General Interrupt | CANIT | | OVRTIM | OVRBUF | SERG | CERG | FERG | AERG |
| CANBT1 | B4h | CAN bit Timing 1 | | BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 | |
| CANBT2 | B5h | CAN bit Timing 2 | | SJW1 | SJW0 | | PRS2 | PRS1 | PRS0 | |
| CANBT3 | B6h | CAN bit Timing 3 | | PHS22 | PHS21 | PHS20 | PHS12 | PHS11 | PHS10 | SMP |

Table 9. CAN SFRs (Continued)

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|--|--------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|
| CANEN | CFh | CAN Enable Channel byte | | | | | ENCH3 | ENCH2 | ENCH1 | ENCH0 |
| CANGIE | C1h | CAN General Interrupt Enable | | | ENRX | ENTX | ENERCH | ENBUF | ENERG | |
| CANIE | C3h | CAN Interrupt Enable Channel byte | | | | | IECH3 | IECH2 | IECH1 | IECH0 |
| CANSIT | BBh | CAN Status Interrupt Channel byte | | | | | SIT3 | SIT2 | SIT1 | SIT0 |
| CANTCON | A1h | CAN Timer Control | TPRESC 7 | TPRESC 6 | TPRESC 5 | TPRESC 4 | TPRESC 3 | TPRESC 2 | TPRESC 1 | TPRESC 0 |
| CANTIMH | ADh | CAN Timer high | CANTIM 15 | CANTIM 14 | CANTIM 13 | CANTIM 12 | CANTIM 11 | CANTIM 10 | CANTIM 9 | CANTIM 8 |
| CANTIML | ACH | CAN Timer low | CANTIM 7 | CANTIM 6 | CANTIM 5 | CANTIM 4 | CANTIM 3 | CANTIM 2 | CANTIM 1 | CANTIM 0 |
| CANSTMPH | AFh | CAN Timer Stamp high | TIMSTMP 15 | TIMSTMP 14 | TIMSTMP 13 | TIMSTMP 12 | TIMSTMP 11 | TIMSTMP 10 | TIMSTMP 9 | TIMSTMP 8 |
| CANSTMPL | Aeh | CAN Timer Stamp low | TIMSTMP 7 | TIMSTMP 6 | TIMSTMP 5 | TIMSTMP 4 | TIMSTMP 3 | TIMSTMP 2 | TIMSTMP 1 | TIMSTMP 0 |
| CANTTCH | A5h | CAN Timer TTC high | TIMTTC 15 | TIMTTC 14 | TIMTTC 13 | TIMTTC 12 | TIMTTC 11 | TIMTTC 10 | TIMTTC 9 | TIMTTC 8 |
| CANTTCL | A4h | CAN Timer TTC low | TIMTTC 7 | TIMTTC 6 | TIMTTC 5 | TIMTTC 4 | TIMTTC 3 | TIMTTC 2 | TIMTTC 1 | TIMTTC 0 |
| CANTEC | 9Ch | CAN Transmit Error Counter | TEC7 | TEC6 | TEC5 | TEC4 | TEC3 | TEC2 | TEC1 | TEC0 |
| CANREC | 9Dh | CAN Receive Error Counter | REC7 | REC6 | REC5 | REC4 | REC3 | REC2 | REC1 | REC0 |
| CANPAGE | B1h | CAN Page | - | - | CHNB1 | CHNB0 | AINC | INDX2 | INDX1 | INDX0 |
| CANSTCH | B2h | CAN Status Channel | DLCW | TXOK | RXOK | BERR | SERR | CERR | FERR | AERR |
| CANCONCH | B3h | CAN Control Channel | CONCH1 | CONCH0 | RPLV | IDE | DLC3 | DLC2 | DLC1 | DLC0 |
| CANMSG | A3h | CAN Message Data | MSG7 | MSG6 | MSG5 | MSG4 | MSG3 | MSG2 | MSG1 | MSG0 |
| CANIDT1 | BCh | CAN Identifier Tag byte 1(Part A) CAN Identifier Tag byte 1(PartB) | IDT10 IDT28 | IDT9 IDT27 | IDT8 IDT26 | IDT7 IDT25 | IDT6 IDT24 | IDT5 IDT23 | IDT4 IDT22 | IDT3 IDT21 |
| CANIDT2 | BDh | CAN Identifier Tag byte 2 (PartA) CAN Identifier Tag byte 2 (PartB) | IDT2 IDT20 | IDT1 IDT19 | IDT0 IDT18 | - IDT17 | - IDT16 | - IDT15 | - IDT14 | - IDT13 |
| CANIDT3 | BEh | CAN Identifier Tag byte 3(PartA) CAN Identifier Tag byte 3(PartB) | - IDT12 | - IDT11 | - IDT10 | - IDT9 | - IDT8 | - IDT7 | - IDT6 | - IDT5 |
| CANIDT4 | BFh | CAN Identifier Tag byte 4(PartA) CAN Identifier Tag byte 4(PartB) | - IDT4 | - IDT3 | - IDT2 | - IDT1 | - IDT0 | RTRTAG | - RB1TAG | - RB0TAG |
| CANIDM1 | C4h | CAN Identifier Mask byte 1(PartA) CAN Identifier Mask byte 1(PartB) | IDMSK10 IDMSK28 | IDMSK9 IDMSK27 | IDMSK8 IDMSK26 | IDMSK7 IDMSK25 | IDMSK6 IDMSK24 | IDMSK5 IDMSK23 | IDMSK4 IDMSK22 | IDMSK3 IDMSK21 |

Table 9. CAN SFRs (Continued)

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|-----------------------------------|---------|---------|---------|---------|---------|---------|---------|---------|
| CANIDM2 | C5h | CAN Identifier Mask byte 2(PartA) | IDMSK2 | IDMSK1 | IDMSK0 | - | - | - | - | - |
| | | CAN Identifier Mask byte 2(PartB) | IDMSK20 | IDMSK19 | IDMSK18 | IDMSK17 | IDMSK16 | IDMSK15 | IDMSK14 | IDMSK13 |
| CANIDM3 | C6h | CAN Identifier Mask byte 3(PartA) | - | - | - | - | - | - | - | - |
| | | CAN Identifier Mask byte 3(PartB) | IDMSK12 | IDMSK11 | IDMSK10 | IDMSK9 | IDMSK8 | IDMSK7 | IDMSK6 | IDMSK5 |
| CANIDM4 | C7h | CAN Identifier Mask byte 4(PartA) | - | - | - | - | - | RTRMSK | - | IDEMSK |
| | | CAN Identifier Mask byte 4(PartB) | IDMSK4 | IDMSK3 | IDMSK2 | IDMSK1 | IDMSK0 | | | |

Table 10. Other SFRs

| Mnemonic | Add | Name | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|-----|----------------------|-------|-------|--------|-------|------|-------|-------|--------|
| PCON | 87h | Power Control | SMOD1 | SMOD0 | | POF | GF1 | GF0 | PD | IDL |
| AUXR1 | A2h | Auxiliary Register 1 | | | ENBOOT | | GF3 | 0 | | DPS |
| CKCON | 8Fh | Clock Control | CANX2 | WDX2 | PCAX2 | SIX2 | T2X2 | T1X2 | T0X2 | X2 |
| FCON | D1h | Flash Control | FPL3 | FPL2 | FPL1 | FPL0 | FPS | FMOD1 | FMOD0 | FBUSY |
| EECON | D2h | EEPROM Contol | EEPL3 | EEPL2 | EEPL1 | EEPL0 | | | EEE | EEBUSY |

Table 11. SFR Mapping

| | 0/8 ⁽¹⁾ | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |
|-----|---------------------------|----------------------|-----------------------------------|-----------------------|----------------------|----------------------|-----------------------|-----------------------|-----|
| F8h | IPL1 xxxx x000 | CH 0000 0000 | CCAP0H 0000 0000 | CCAP1H 0000 0000 | | | | | FFh |
| F0h | B 0000 0000 | | ADCLK xxx0 0000 | ADCON x000 0000 | ADDL 0000 0000 | ADDH 0000 0000 | ADCF 0000 0000 | IPH1 xxxx x000 | F7h |
| E8h | IEN1 xxxx x000 | CL 0000 0000 | CCAP0L 0000 0000 | CCAP1L 0000 0000 | | | | | EFh |
| E0h | ACC 0000 0000 | | | | | | | | E7h |
| D8h | CCON 0000 0000 | CMOD 0xxx x000 | CCAPM0 x000 0000 | CCAPM1 x000 0000 | | | | | DFh |
| D0h | PSW 0000 0000 | FCON 0000 0000 | EECON xxxx xx00 | | | | | | D7h |
| C8h | T2CON 0000 0000 | T2MOD xxxx xx00 | RCAP2L 0000 0000 | RCAP2H 0000 0000 | TL2 0000 0000 | TH2 0000 0000 | | CANEN xxxx 0000 | CFh |
| C0h | P4 xxxx xx11 | CANGIE 1100 0000 | | CANIE 1111 0000 | CANIDM1 xxxx xxxx | CANIDM2 xxxx xxxx | CANIDM3 xxxx xxxx | CANIDM4 xxxx xxxx | C7h |
| B8h | IPL0 x000 0000 | SADEN 0000 0000 | | CANSIT xxxx 0000 | CANIDT1 xxxx xxxx | CANIDT2 xxxx xxxx | CANIDT3 xxxx xxxx | CANIDT4 xxxx xxxx | BFh |
| B0h | P3 1111 1111 | CANPAGE 1100 0000 | CANSTCH xxxx xxxx | CANCONCH xxxx xxxx | CANBT1 xxxx xxxx | CANBT2 xxxx xxxx | CANBT3 xxxx xxxx | IPH0 x000 0000 | B7h |
| A8h | IEN0 0000 0000 | SADDR 0000 0000 | CANGSTA 1010 0000 | CANGCON 0000 0000 | CANTIML 0000 0000 | CANTIMH 0000 0000 | CANSTMPL xxxx xxxx | CANSTMPH xxxx xxxx | AFh |
| A0h | P2 xxxx xx11 | CANTCON 0000 0000 | AUXR1 ⁽²⁾ xxxx 00x0 | CANMSG xxxx xxxx | CANTTCL 0000 0000 | CANTTCH 0000 0000 | WDTRST 1111 1111 | WDTPRG xxxx x000 | A7h |
| 98h | SCON 0000 0000 | SBUF 0000 0000 | | CANGIT 0x00 0000 | CANTEC 0000 0000 | CANREC 0000 0000 | | | 9Fh |
| 90h | P1 1111 1111 | | | | | | | | 97h |
| 88h | TCON 0000 0000 | TMOD 0000 0000 | TL0 0000 0000 | TL1 0000 0000 | TH0 0000 0000 | TH1 0000 0000 | | CKCON 0000 0000 | 8Fh |
| 80h | | SP 0000 0111 | DPL 0000 0000 | DPH 0000 0000 | | | | PCON 00x1 0000 | 87h |
| | 0/8 ⁽¹⁾ | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |

Reserved

Notes: 1. These registers are bit-addressable.

Sixteen addresses in the SFR space are both byte-addressable and bit-addressable. The bit-addressable SFRs are those whose address ends in 0 and 8. The bit addresses, in this area, are 0x80 through to 0xFF.

2. AUXR1 bit ENBOOT is initialized with the content of the BLJB bit inverted.

Clock

The T89C51CC02 core needs only 6 clock periods per machine cycle. This feature, called “X2”, provides the following advantages:

- Divides frequency crystals by 2 (cheaper crystals) while keeping the same CPU power.
- Saves power consumption while keeping the same CPU power (oscillator power saving).
- Saves power consumption by dividing dynamic operating frequency by 2 in operating and idle modes.
- Increases CPU power by 2 while keeping the same crystal frequency.

In order to keep the original C51 compatibility, a divider-by-2 is inserted between the XTAL1 signal and the main clock input of the core (phase generator). This divider may be disabled by the software.

An extra feature is available to start after Reset in the X2 Mode. This feature can be enabled by a bit X2B in the Hardware Security Byte. This bit is described in the section ‘In-System Programming’.

Description

The X2 bit in the CKCON register (See Table 12) allows switching from 12 clock cycles per instruction to 6 clock cycles and vice versa. At reset, the standard speed is activated (STD mode).

Setting this bit activates the X2 feature (X2 Mode) for the CPU Clock only (See Figure 3).

The Timers 0, 1 and 2, Uart, PCA, watchdog or CAN switch in X2 Mode only if the corresponding bit is cleared in the CKCON register.

The clock for the whole circuit and peripheral is first divided by two before being used by the CPU core and peripherals. This allows any cyclic ratio to be accepted on the XTAL1 input. In X2 Mode, as this divider is bypassed, the signals on XTAL1 must have a cyclic ratio between 40 to 60%. Figure 3. shows the clock generation block diagram. The X2 bit is validated on the $XTAL1 \div 2$ rising edge to avoid glitches when switching from the X2 to the STD mode. Figure 4 shows the mode switching waveforms.

Figure 3. Clock CPU Generation Diagram

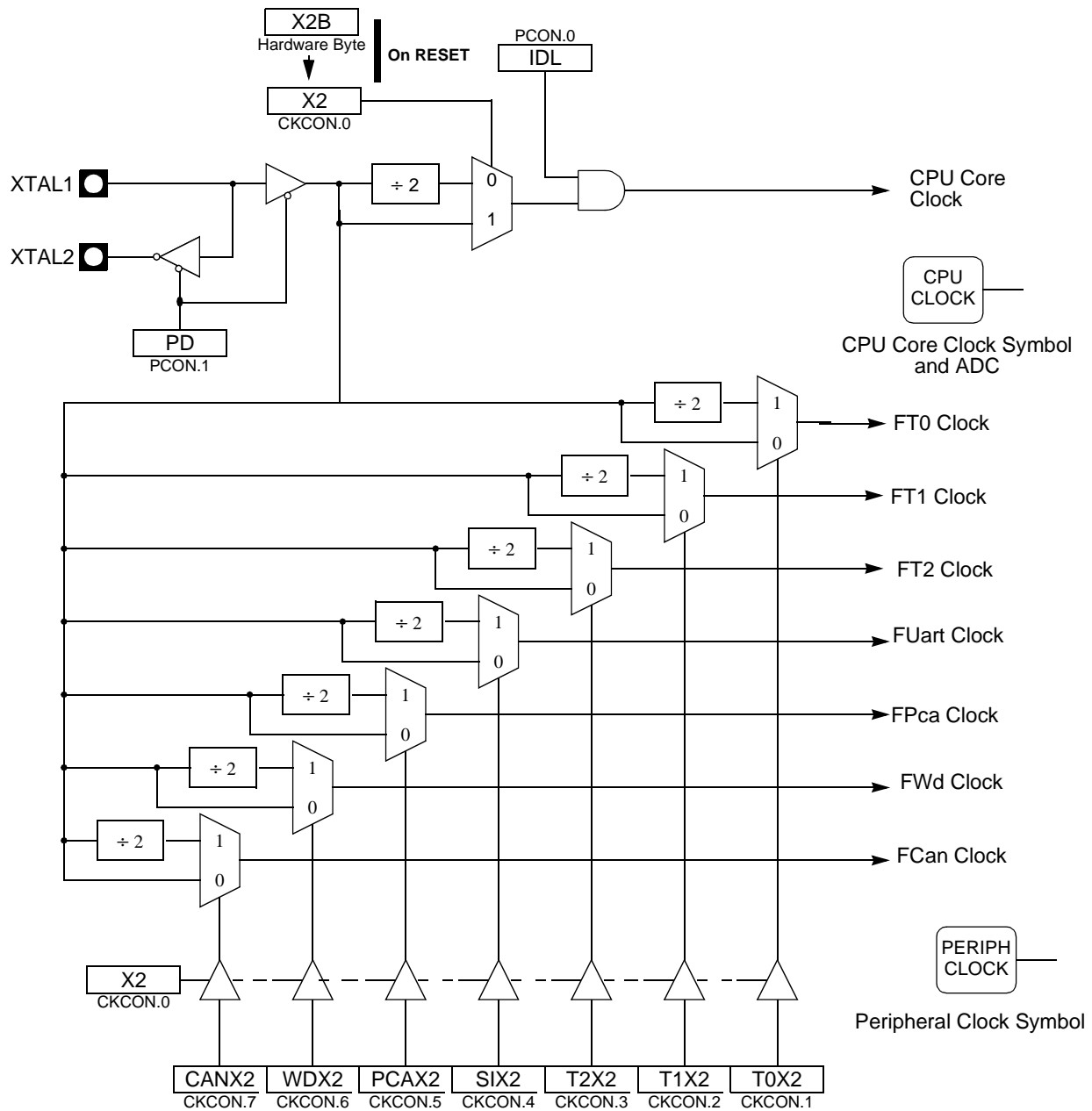
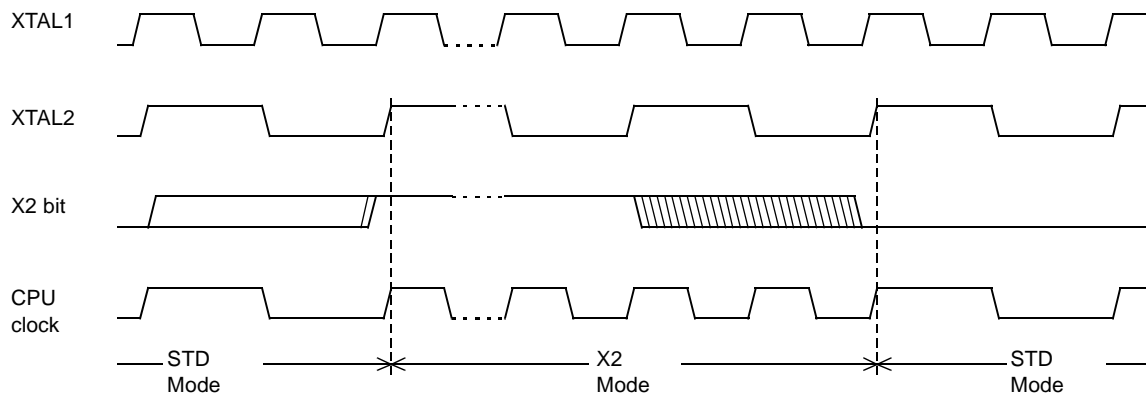


Figure 4. Mode Switching Waveforms⁽¹⁾



Note: 1. In order to prevent any incorrect operation while operating in the X2 Mode, users must be aware that all peripherals using the clock frequency as a time reference (UART, timers...) will have their time reference divided by 2. For example, a free running timer generating an interrupt every 20 ms will then generate an interrupt every 10 ms. A UART with a 4800 baud rate will have a 9600 baud rate.

Register

Table 12. CKCON Register
CKCON (S:8Fh)
Clock Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|------|------|------|------|----|
| CANX2 | WDX2 | PCAX2 | SIX2 | T2X2 | T1X2 | T0X2 | X2 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | CANX2 | CAN Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 6 | WDX2 | Watchdog Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 5 | PCAX2 | Programmable Counter Array Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 4 | SIX2 | Enhanced UART clock (MODE 0 and 2) ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 3 | T2X2 | Timer 2 Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 2 | T1X2 | Timer 1 Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 1 | T0X2 | Timer 0 Clock ⁽¹⁾ Clear to select 6 clock periods per peripheral clock cycle. Set to select 12 clock periods per peripheral clock cycle. | | | | | |
| 0 | X2 | CPU Clock Clear to select 12 clock periods per machine cycle (STD mode) for CPU and all the peripherals. Set to select 6 clock periods per machine cycle (X2 Mode) and to enable the individual peripherals 'X2' bits. | | | | | |

Note: 1. This control bit is validated when the CPU clock bit X2 is set; when X2 is low, this bit has no effect.

Reset Value = 0000 0000b

Power Management

Two power reduction modes are implemented in the A/T89C51CC02: the Idle mode and the Power-down mode. These modes are detailed in the following sections. In addition to these power reduction modes, the clocks of the core and peripherals can be dynamically divided by 2 using the X2 Mode detailed in Section “Clock”.

Reset Pin

In order to start-up (cold reset) or to restart (warm reset) properly the microcontroller, a high level has to be applied on the RST pin. A bad level leads to a wrong initialisation of the internal registers like SFRs, PC, etc. and to unpredictable behavior of the microcontroller. A warm reset can be applied either directly on the RST pin or indirectly by an internal reset source such as a watchdog, PCA, timer, etc.

At Power-up (cold reset)

Two conditions are required before enabling a CPU start-up:

- VDD must reach the specified VDD range,
- The level on xtal1 input must be outside the specification (VIH, VIL).

If one of these two conditions are not met, the microcontroller does not start correctly and can execute an instruction fetch from anywhere in the program space. An active level applied on the RST pin must be maintained until both of the above conditions are met. A reset is active when the level VIH1 is reached and when the pulse width covers the period of time where VDD and the oscillator are not stabilized. Two parameters have to be taken into account to determine the reset pulse width:

- VDD rise time (vddrst),
- Oscillator startup time (oscrst).

To determine the capacitor the highest value of these two parameters has to be chosen. The reset circuitry is shown in Figure 5.

Figure 5. Reset Circuitry

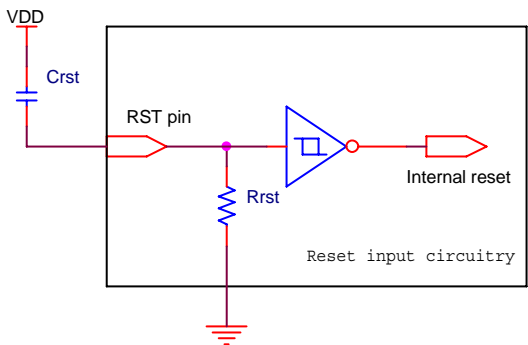


Table 13 and Table 14 give some typical examples for three values of VDD rise times, two values of oscillator start-up time and two pull-down resistor values.

Table 13. Minimum Reset Capacitor for a 50K Pull-down Resistor

| oscrst/vddrst | 1ms | 10ms | 100ms |
|---------------|-------|-------|-------|
| 5ms | 820nF | 1.2µF | 12µF |
| 20ms | 2.7µF | 3.9µF | 12µF |

Table 14. Minimum Reset Capacitor for a 15k Pull-down Resistor

| oscrst/vddrst | 1ms | 10ms | 100ms |
|---------------|-------------|-------------|------------|
| 5ms | 2.7 μ F | 4.7 μ F | 47 μ F |
| 20ms | 10 μ F | 15 μ F | 47 μ F |

Note: These values assume VDD starts from 0v to the nominal value. If the time between two on/off sequences is too fast, the power-supply decoupling capacitors may not be fully discharged, leading to a bad reset sequence.

During a Normal Operation (Warm Reset)

Reset pin must be maintained for at least 2 machine cycles (24 oscillator clock periods) to apply a reset sequence during normal operation. The number of clock periods is mode independent (X2 or X1).

Watchdog Reset

A 1K resistor must be added in series with the capacitor to allow the use of watchdog reset pulse output on the RST pin or when an external power-supply supervisor is used. Figure 6 shows the reset circuitry when a capacitor is used.

Figure 6. Reset Circuitry for a Watchdog Configuration

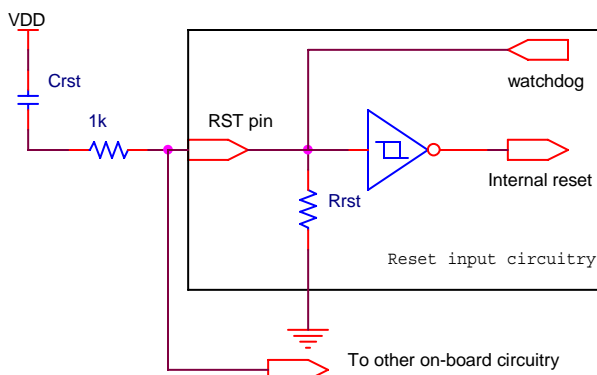
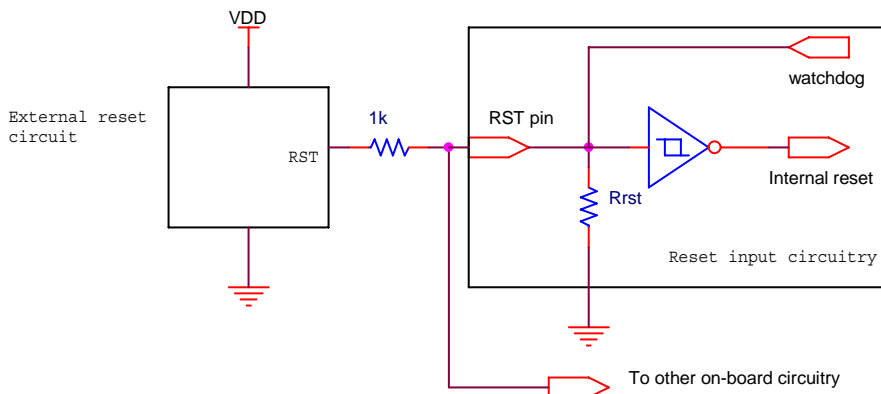


Figure 7 shows the reset circuitry when an external reset circuit is used.

Figure 7. Reset Circuitry Example Using an External Reset Circuit



Reset Recommendation to Prevent Flash Corruption

When a Flash program memory is embedded on-chip, it is strongly recommended to use an external reset chip (brown out device) to apply a reset (Figure 7). It prevents system malfunction during periods of insufficient power-supply voltage (power-supply failure, power supply switched off, etc.).

Idle Mode

Idle mode is a power reduction mode that reduces the power consumption. In this mode, program execution halts. Idle mode freezes the clock to the CPU at known states while the peripherals continue to be clocked. The CPU status before entering Idle mode is preserved, i.e., the program counter and program status word register retain their data for the duration of Idle mode. The contents of the SFRs and RAM are also retained. The status of the Port pins during Idle mode is detailed in Table 15.

Entering Idle Mode

To enter Idle mode, set the IDL bit in PCON register (See Table 16). The A/T89C51CC02 enters Idle mode upon execution of the instruction that sets IDL bit. The instruction that sets IDL bit is the last instruction executed.

Note: If IDL bit and PD bit are set simultaneously, the A/T89C51CC02 enters Power-down mode. Then it does not go in Idle mode when exiting Power-down mode.

Exiting Idle Mode

There are two ways to exit Idle mode:

1. Generate an enabled interrupt.

Hardware clears IDL bit in PCON register which restores the clock to the CPU. Execution resumes with the interrupt service routine. Upon completion of the interrupt service routine, program execution resumes with the instruction immediately following the instruction that activated Idle mode. The general purpose flags (GF1 and GF0 in PCON register) may be used to indicate whether an interrupt occurred during normal operation or during Idle mode. When Idle mode is exited by an interrupt, the interrupt service routine may examine GF1 and GF0.

2. Generate a reset.

A logic high on the RST pin clears IDL bit in PCON register directly and asynchronously. This restores the clock to the CPU. Program execution momentarily resumes with the instruction immediately following the instruction that activated the Idle mode and may continue for a number of clock cycles before the internal reset algorithm takes control. Reset initializes the A/T89C51CC02 and vectors the CPU to address C:0000h.

- Notes:
1. During the time that execution resumes, the internal RAM cannot be accessed; however, it is possible for the Port pins to be accessed. To avoid unexpected outputs at the Port pins, the instruction immediately following the instruction that activated Idle mode should not write to a Port pin or to the external RAM.
 2. If Idle mode is invoked by ADC Idle, the ADC conversion completion will exit Idle.

Power-down Mode

The Power-down mode places the A/T89C51CC02 in a very low power state. Power-down mode stops the oscillator, freezes all clock at known states. The CPU status prior to entering Power-down mode is preserved, i.e., the program counter, program status word register retain their data for the duration of Power-down mode. In addition, the SFRs and RAM contents are preserved. The status of the Port pins during Power-down mode is detailed in Table 15.

Entering Power-down Mode

To enter Power-down mode, set PD bit in PCON register. The A/T89C51CC02 enters the Power-down mode upon execution of the instruction that sets PD bit. The instruction that sets PD bit is the last instruction executed.

Exiting Power-down Mode

Note: If V_{DD} was reduced during the Power-down mode, do not exit Power-down mode until V_{DD} is restored to the normal operating level.

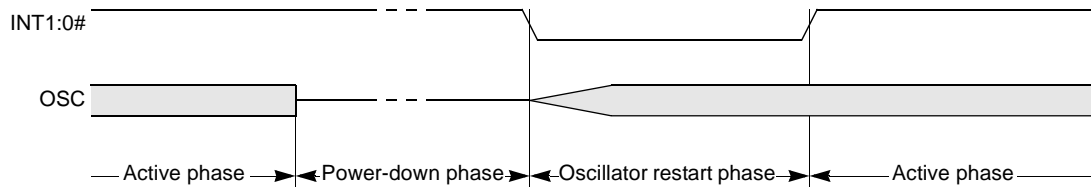
There are two ways to exit the Power-down mode:

1. Generate an enabled external interrupt.
 - The A/T89C51CC02 provides capability to exit from Power-down using INT0#, INT1#.

Hardware clears PD bit in PCON register which starts the oscillator and restores the clocks to the CPU and peripherals. Using INTx# input, execution resumes when the input is released (See Figure 8). Execution resumes with the interrupt service routine. Upon completion of the interrupt service routine, program execution resumes with the instruction immediately following the instruction that activated Power-down mode.

- Notes:
1. The external interrupt used to exit Power-down mode must be configured as level sensitive (INT0# and INT1#) and must be assigned the highest priority. In addition, the duration of the interrupt must be long enough to allow the oscillator to stabilize. The execution will only resume when the interrupt is deasserted.
 2. Exit from power-down by external interrupt does not affect the SFRs nor the internal RAM content.

Figure 8. Power-down Exit Waveform Using INT1:0#



2. Generate a reset.
 - A logic high on the RST pin clears PD bit in PCON register directly and asynchronously. This starts the oscillator and restores the clock to the CPU and peripherals. Program execution momentarily resumes with the instruction immediately following the instruction that activated Power-down mode and may continue for a number of clock cycles before the internal reset algorithm takes control. Reset initializes the A/T89C51CC02 and vectors the CPU to address 0000h.

- Notes:
1. During the time that execution resumes, the internal RAM cannot be accessed; however, it is possible for the Port pins to be accessed. To avoid unexpected outputs at the Port pins, the instruction immediately following the instruction that activated the Power-down mode should not write to a Port pin or to the external RAM.
 2. Exit from power-down by reset redefines all the SFRs, but does not affect the internal RAM content.

Table 15. Pin Conditions in Special Operating Modes

| Mode | Port 1 | Port 2 | Port 3 | Port 4 |
|--------------------------------------|--------|--------|--------|--------|
| Reset | High | High | High | High |
| Idle (internal code) | Data | Data | Data | Data |
| Idle (external code) | Data | Data | Data | Data |
| Power- Down(inter- nal code) | Data | Data | Data | Data |
| Power- Down (external code) | Data | Data | Data | Data |

Registers

Table 16. PCON Register
PCON (S:87h)
Power Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-----|-----|-----|----|-----|
| SMOD1 | SMOD0 | - | POF | GF1 | GF0 | PD | IDL |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | SMOD1 | Serial port Mode bit 1 Set to select double baud rate in mode 1, 2 or 3. | | | | | |
| 6 | SMOD0 | Serial port Mode bit 0 Clear to select SM0 bit in SCON register. Set to select FE bit in SCON register. | | | | | |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 4 | POF | Power-off Flag Clear to recognize next reset type. Set by hardware when V_{CC} rises from 0 to its nominal voltage. Can also be set by software. | | | | | |
| 3 | GF1 | General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage. | | | | | |
| 2 | GF0 | General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage. | | | | | |
| 1 | PD | Power-down Mode bit Cleared by hardware when reset occurs. Set to enter power-down mode. | | | | | |
| 0 | IDL | Idle Mode bit Clear by hardware when interrupt or reset occurs. Set to enter idle mode. | | | | | |

Reset Value = 00X1 0000b

Not bit addressable

Data Memory

The T89C51CC02 provides data memory access in two different spaces:

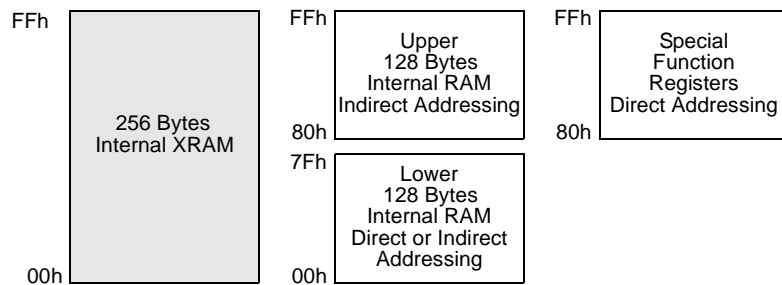
The internal space mapped in three separate segments:

- The lower 128 Bytes RAM segment.
- The upper 128 Bytes RAM segment.
- The expanded 256 Bytes RAM segment (XRAM).

A fourth internal segment is available but dedicated to Special Function Registers, SFRs, (addresses 80h to FFh) accessible by direct addressing mode.

Figure 9 shows the internal data memory spaces organization.

Figure 9. Internal memory - RAM



Internal Space

Lower 128 Bytes RAM

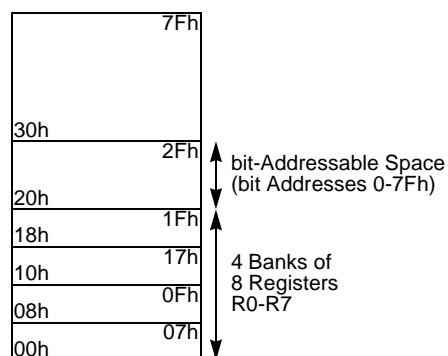
The lower 128 Bytes of RAM (See Figure 10) are accessible from address 00h to 7Fh using direct or indirect addressing modes. The lowest 32 Bytes are grouped into 4 banks of 8 registers (R0 to R7). Two bits RS0 and RS1 in PSW register (See Table 18) select which bank is in use according to Table 17. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing, and can be used for context switching in interrupt service routines.

Table 17. Register Bank Selection

| RS1 | RS0 | Description |
|-----|-----|---------------------------------|
| 0 | 0 | Register bank 0 from 00h to 07h |
| 0 | 1 | Register bank 0 from 08h to 0Fh |
| 1 | 0 | Register bank 0 from 10h to 17h |
| 1 | 1 | Register bank 0 from 18h to 1Fh |

The next 16 Bytes above the register banks form a block of bit-addressable memory space. The C51 instruction set includes a wide selection of singlebit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00h to 7Fh.

Figure 10. Lower 128 Bytes Internal RAM Organization



Upper 128 Bytes RAM

The upper 128 Bytes of RAM are accessible from address 80h to FFh using only indirect addressing mode.

Expanded RAM

The on-chip 256 Bytes of expanded RAM (XRAM) are accessible from address 0000h to 00FFh using indirect addressing mode through MOVX instructions. In this address range.

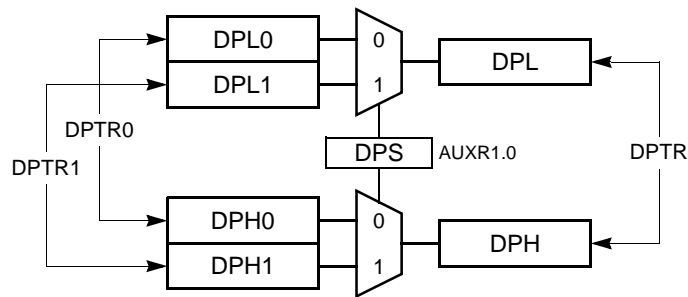
Note: Lower 128 Bytes RAM, Upper 128 Bytes RAM, and expanded RAM are made of volatile memory cells. This means that the RAM content is indeterminate after power-up and must then be initialized properly.

Dual Data Pointer

Description

The T89C51CC02 implements a second data pointer for speeding up code execution and reducing code size in case of intensive usage of external memory accesses. DPTR0 and DPTR1 are seen by the CPU as DPTR and are accessed using the SFR addresses 83h and 84h that are the DPH and DPL addresses. The DPS bit in AUXR1 register (See Figure 19) is used to select whether DPTR is the data pointer 0 or the data pointer 1 (See Figure 11).

Figure 11. Dual Data Pointer Implementation



Application

Software can take advantage of the additional data pointers to both increase speed and reduce code size, for example, block operations (copy, compare...) are well served by using one data pointer as a "source" pointer and the other one as a "destination" pointer. Hereafter is an example of block move implementation using the two pointers and coded in assembler. The latest C compiler takes also advantage of this feature by providing enhanced algorithm libraries.

The INC instruction is a short (2 Bytes) and fast (6 machine cycle) way to manipulate the DPS bit in the AUXR1 register. However, note that the INC instruction does not directly force the DPS bit to a particular state, but simply toggles it. In simple routines, such as the block move example, only the fact that DPS is toggled in the proper sequence matters, not its actual value. In other words, the block move routine works the same whether DPS is 0 or 1 on entry.

```
; ASCII block move using dual data pointers
; Modifies DPTR0, DPTR1, A and PSW
; Ends when encountering NULL character
; Note: DPS exits opposite to the entry state unless an extra INC AUXR1 is
added
```

```
AUXR1 EQU 0A2h
```

```
move: mov DPTR, #SOURCE ; address of SOURCE
      inc AUXR1 ; switch data pointers
      mov DPTR, #DEST ; address of DEST
mv_loop: inc AUXR1 ; switch data pointers
      movx A, @DPTR ; get a byte from SOURCE
      inc DPTR ; increment SOURCE address
      inc AUXR1 ; switch data pointers
      movx @DPTR, A ; write the byte to DEST
      inc DPTR ; increment DEST address
      jnz mv_loop ; check for NULL terminator
end_move:
```

Registers

Table 18. PSW Register
PSW (S:D0h)
Program Status Word Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-----|-----|----|----|---|
| CY | AC | F0 | RS1 | RS0 | OV | F1 | P |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | CY | Carry Flag Carry out from bit 1 of ALU operands. | | | | | |
| 6 | AC | Auxiliary Carry Flag Carry out from bit 1 of addition operands. | | | | | |
| 5 | F0 | User Definable Flag 0 | | | | | |
| 4 - 3 | RS1:0 | Register Bank Select bits Refer to Table 17 for bits description. | | | | | |
| 2 | OV | Overflow Flag Overflow set by arithmetic operations. | | | | | |
| 1 | F1 | User Definable Flag 1 | | | | | |
| 0 | P | Parity bit Set when ACC contains an odd number of 1's. Cleared when ACC contains an even number of 1's. | | | | | |

Reset Value = 0000 0000b

Table 19. AUXR1 Register
AUXR1 (S:A2h)
Auxiliary Control Register 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|--------|---|-----|---|---|-----|
| - | - | ENBOOT | - | GF3 | 0 | - | DPS |

| Bit Number | Bit Mnemonic | Description |
|------------|-----------------------|---|
| 7 - 6 | - | Reserved The value read from these bits is indeterminate. Do not set these bits. |
| 5 | ENBOOT ⁽¹⁾ | Enable Boot Flash Set this bit to map the boot Flash between F800h -FFFFh Clear this bit to disable boot Flash. |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 | GF3 | General Purpose Flag 3 |
| 2 | 0 | Always Zero This bit is stuck to logic 0 to allow INC AUXR1 instruction without affecting GF3 flag. |
| 1 | - | Reserved for Data Pointer Extension |
| 0 | DPS | Data Pointer Select bit Set to select second dual data pointer: DPTR1. Clear to select first dual data pointer: DPTR0. |

Reset Value = XXXX 00X0b

Note: 1. ENBOOT is initialized with the invert BLJB at reset. See In-System Programming section.

EEPROM Data Memory

The 2K bytes on-chip EEPROM memory block is located at addresses 0000h to 07FFh of the XRAM/XRAM memory space and is selected by setting control bits in the EECON register. A read in the EEPROM memory is done with a MOVX instruction.

A physical write in the EEPROM memory is done in two steps: write data in the column latches and transfer of all data latches into an EEPROM memory row (programming).

The number of data written on the page may vary from 1 up to 128 Bytes (the page size). When programming, only the data written in the column latch is programmed and a ninth bit is used to obtain this feature. This provides the capability to program the whole memory by Bytes, by page or by a number of Bytes in a page. Indeed, each ninth bit is set when the writing the corresponding byte in a row and all these ninth bits are reset after the writing of the complete EEPROM row.

Write Data in the Column Latches

Data is written by byte to the column latches as for an external RAM memory. Out of the 11 address bits of the data pointer, the 4 MSBs are used for page selection (row) and 7 are used for byte selection. Between two EEPROM programming sessions, all the addresses in the column latches must stay on the same page, meaning that the 4 MSB must no be changed.

The following procedure is used to write to the column latches:

- Save and disable interrupt
- Set bit EEE of EECON register
- Load DPTR with the address to write
- Store A register with the data to be written
- Execute a MOVX @DPTR, A
- If needed loop the three last instructions until the end of a 128 Bytes page
- Restore interrupt

Note: The last page address used when loading the column latch is the one used to select the page programming address.

Programming

The EEPROM programming consists of the following actions:

- Write one or more Bytes of one page in the column latches. Normally, all Bytes must belong to the same page; if not, the last page address will be latched and the others discarded.
- Launch programming by writing the control sequence (50h followed by A0h) to the EECON register.
- EEBUSY flag in EECON is then set by hardware to indicate that programming is in progress and that the EEPROM segment is not available for reading.
- The end of programming is indicated by a hardware clear of the EEBUSY flag.

Note: The sequence 5xh and Axh must be executed without instructions between then otherwise the programming is aborted.

Read Data

The following procedure is used to read the data stored in the EEPROM memory:

- Save and disable interrupt
- Set bit EEE of EECON register
- Load DPTR with the address to read
- Execute a MOVX A, @DPTR
- Restore interrupt

Examples

```

;F*****
;* NAME: api_rd_eeprom_byte
;* DPTR contain address to read.
;* Acc contain the reading value
;* NOTE: before execute this function, be sure the EEPROM is not BUSY
;*****
api_rd_eeprom_byte:
; Save and clear EA
MOV    EECON, #02h; map EEPROM in XRAM space
MOVX   A, @DPTR
MOV    EECON, #00h; unmap EEPROM
; Restore EA
ret

;F*****
;* NAME: api_ld_eeprom_cl
;* DPTR contain address to load
;* Acc contain value to load
;* NOTE: in this example we load only 1 byte, but it is possible upto
;* 128 Bytes.
;* before execute this function, be sure the EEPROM is not BUSY
;*****
api_ld_eeprom_cl:
; Save and clear EA
MOV    EECON, #02h ; map EEPROM in XRAM space
MOVX   @DPTR, A
MOVEECON, #00h; unmap EEPROM
; Restore EA
ret

;F*****
;* NAME: api_wr_eeprom
;* NOTE: before execute this function, be sure the EEPROM is not BUSY
;*****
api_wr_eeprom:
; Save and clear EA
MOV    EECON, #050h
MOV    EECON, #0A0h
; Restore EA
ret

```


Registers

Table 20. EECON Register
EECON (S:0D2h)
EEPROM Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-------|---|---|-----|--------|
| EEPL3 | EEPL2 | EEPL1 | EEPL0 | - | - | EEE | EEBUSY |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 4 | EEPL3-0 | Programming Launch Command bits Write 5Xh followed by AXh to EEPL to launch the programming. | | | | | |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 2 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 1 | EEE | Enable EEPROM Space bit Set to map the EEPROM space during MOVX instructions (Write in the column latches) Clear to map the XRAM space during MOVX. | | | | | |
| 0 | EEBUSY | Programming Busy Flag Set by hardware when programming is in progress. Cleared by hardware when programming is done. Can not be set or cleared by software. | | | | | |

Reset Value = XXXX XX00b

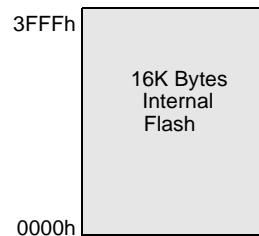
Not bit addressable

Program/Code Memory

The T89C51CC02 implement 16K Bytes of on-chip program/code memory.

The Flash memory increases EPROM and ROM functionality by in-circuit electrical erasure and programming. Thanks to the internal charge pump, the high voltage needed for programming or erasing Flash cells is generated on-chip using the standard V_{DD} voltage. Thus, the Flash memory can be programmed using only one voltage and allows In-System Programming (ISP). Hardware programming mode is also available using specific programming tool.

Figure 12. Program/Code Memory Organization



Flash Memory Architecture

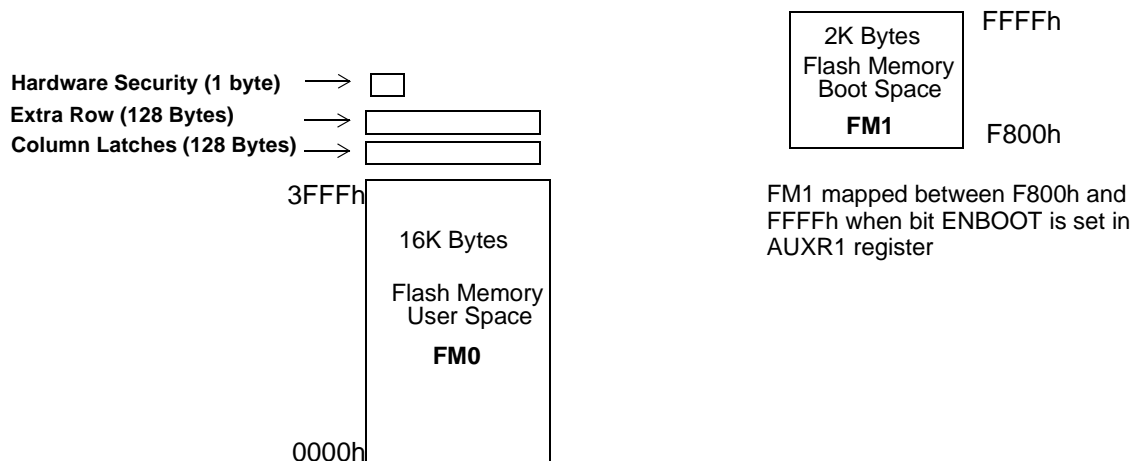
T89C51CC02 features two on-chip Flash memories:

- Flash memory FM0:
containing 16K Bytes of program memory (user space) organized into 128 bytes pages,
- Flash memory FM1:
2K Bytes for boot loader and Application Programming Interfaces (API).

The FM0 can be program by both parallel programming and Serial ISP whereas FM1 supports only parallel programming by programmers. The ISP mode is detailed in the 'In-System Programming' section.

All Read/Write access operations on Flash memory by user application are managed by a set of API described in the 'In-System Programming' section.

Figure 13. Flash Memory Architecture



FM0 Memory Architecture

The Flash memory is made up of 4 blocks (See Figure 13):

1. The memory array (user space) 16K Bytes
2. The Extra Row
3. The Hardware security bits
4. The column latch registers

User Space

This space is composed of a 16K Bytes Flash memory organized in 128 pages of 128 Bytes. It contains the user's application code.

Extra Row (XRow)

This row is a part of FM0 and has a size of 128 Bytes. The extra row may contain information for boot loader usage.

Hardware Security Byte

The Hardware security Byte space is a part of FM0 and has a size of 1 byte. The 4 MSB can be read/written by software, the 4 LSB can only be read by software and written by hardware in parallel mode.

Column Latches

The column latches, also part of FM0, have a size of full page (128 Bytes). The column latches are the entrance buffers of the three previous memory locations (user array, XROW and Hardware security byte).

Cross Flash Memory Access Description

The FM0 memory can be programmed as describe on Table 21. Programming FM0 from FM0 is impossible.

The FM1 memory can be program only by parallel programming.

Table 21 show all software Flash access allowed.

Table 21. Cross Flash Memory Access

| Code executing from | | Action | FM0 (user Flash) | FM1 (boot Flash) |
|---------------------|---------------------|-------------------|---------------------|---------------------|
| | FM0 (user Flash) | Read | ok | - |
| | | Load column latch | ok | - |
| | | Write | - | - |
| | FM1 (boot Flash) | Read | ok | ok |
| | | Load column latch | ok | - |
| | | Write | ok | - |

Overview of FM0 Operations

The CPU interfaces the Flash memory through the FCON register and AUXR1 register.

These registers are used to:

- Map the memory spaces in the adressable space
- Launch the programming of the memory spaces
- Get the status of the Flash memory (busy/not busy)

Mapping of the Memory Space

By default, the user space is accessed by MOVC instruction for read only. The column latches space is made accessible by setting the FPS bit in FCON register. Writing is possible from 0000h to 3FFFh, address bits 6 to 0 are used to select an address within a page while bits 14 to 7 are used to select the programming address of the page. Setting FPS bit takes precedence on the EEE bit in EECON register.

The other memory spaces (user, extra row, hardware security) are made accessible in the code segment by programming bits FMOD0 and FMOD1 in FCON register in accordance with Table 22. A MOVC instruction is then used for reading these spaces.

Table 22. FM0 blocks Select bits

| FMOD1 | FMOD0 | FM0 Adressable Space |
|-------|-------|--------------------------------|
| 0 | 0 | User (0000h-3FFFh) |
| 0 | 1 | Extra Row(FF80h-FFFFh) |
| 1 | 0 | Hardware Security Byte (0000h) |
| 1 | 1 | Reserved |

Launching Programming

FPL3:0 bits in FCON register are used to secure the launch of programming. A specific sequence must be written in these bits to unlock the write protection and to launch the programming. This sequence is 5xh followed by Axx. Table 23 summarizes the memory spaces to program according to FMOD1:0 bits.

Table 23. Programming Spaces

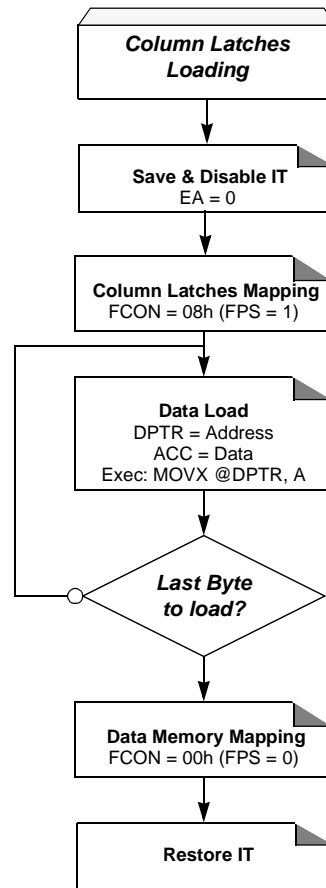
| | Write to FCON | | | | Operation |
|------------------------|---------------|-----|-------|-------|---|
| | FPL3:0 | FPS | FMOD1 | FMOD0 | |
| User | 5 | x | 0 | 0 | No action |
| | A | x | 0 | 0 | Write the column latches in user space |
| Extra Row | 5 | x | 0 | 1 | No action |
| | A | x | 0 | 1 | Write the column latches in extra row space |
| Hardware Security Byte | 5 | x | 1 | 0 | No action |
| | A | x | 1 | 0 | Write the fuse bits space |
| Reserved | 5 | x | 1 | 1 | No action |
| | A | x | 1 | 1 | No action |

Note: The sequence 5xh and Axx must be executing without instructions between them otherwise the programming is aborted.

Interrupts that may occur during programming time must be disabled to avoid any spurious exit of the programming mode.

| | |
|-----------------------------------|--|
| Status of the Flash Memory | <p>The bit FBUSY in FCON register is used to indicate the status of programming. FBUSY is set when programming is in progress.</p> |
| Selecting FM1 | <p>The bit ENBOOT in AUXR1 register is used to map FM1 from F800h to FFFFh.</p> |
| Loading the Column Latches | <p>Any number of data from 1 byte to 128 Bytes can be loaded in the column latches. This provides the capability to program the whole memory by byte, by page or by any number of Bytes in a page.</p> <p>When programming is launched, an automatic erase of the locations loaded in the column latches is first performed, then programming is effectively done. Thus no page or block erase is needed and only the loaded data are programmed in the corresponding page.</p> <p>The following procedure is used to load the column latches and is summarized in Figure 14:</p> <ul style="list-style-type: none"> • Save then disable interrupt and map the column latch space by setting FPS bit. • Load the DPTR with the address to load. • Load Accumulator register with the data to load. • Execute the MOVX @DPTR, A instruction. • If needed loop the three last instructions until the page is completely loaded. • unmap the column latch and Restore Interrupt |

Figure 14. Column Latches Loading Procedure⁽¹⁾



Note: 1. The last page address used when loading the column latch is the one used to select the page programming address.

Programming the Flash Spaces

User

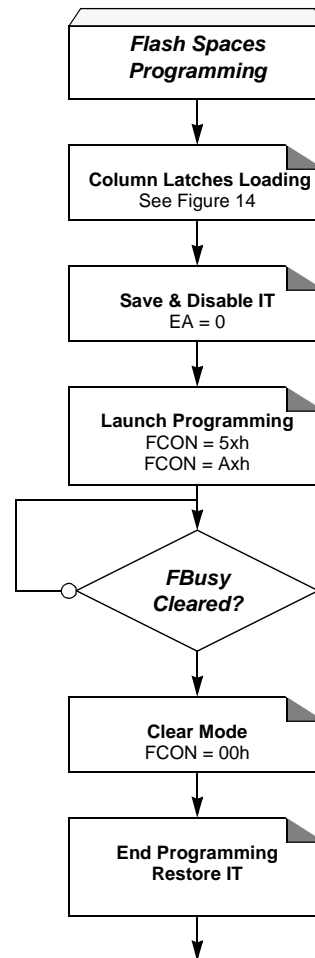
The following procedure is used to program the User space and is summarized in Figure 15:

- Load up to one page of data in the column latches from address 0000h to 3FFFh.
- Save then disable the interrupts.
- Launch the programming by writing the data sequence 50h followed by A0h in FCON register. This step must be executed from FM1.
The end of the programming indicated by the FBUSY flag cleared.
- Restore the interrupts.

Extra Row

The following procedure is used to program the Extra Row space and is summarized in Figure 15:

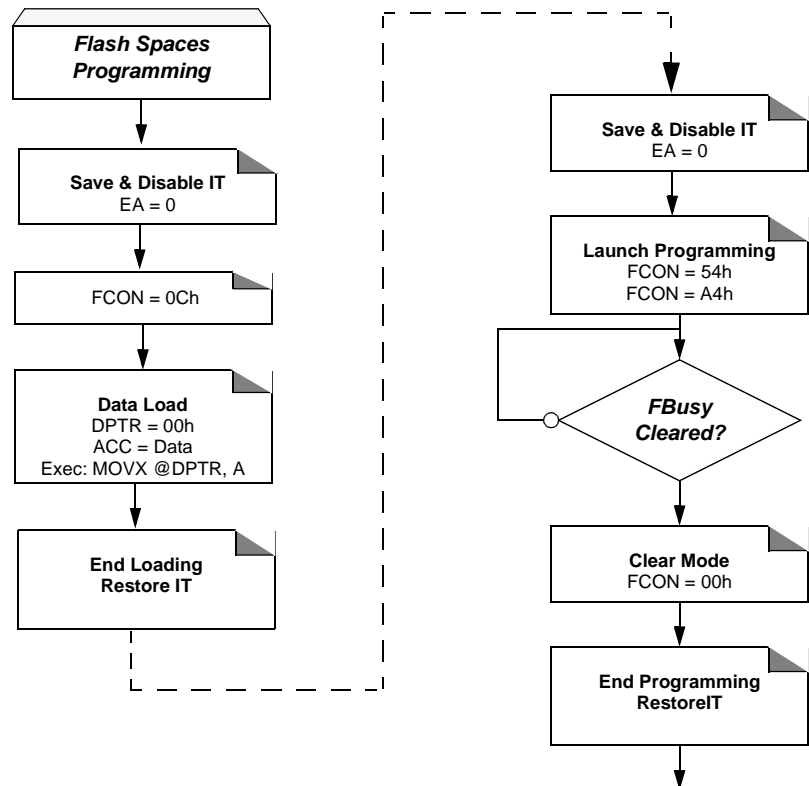
- Load data in the column latches from address FF80h to FFFFh.
- Save then disable the interrupts.
- Launch the programming by writing the data sequence 52h followed by A2h in FCON register. This step of the procedure must be executed from FM1.
The end of the programming indicated by the FBUSY flag cleared.
- Restore the interrupts.

Figure 15. Flash and Extra row Programming Procedure*Hardware Security Byte*

The following procedure is used to program the Hardware Security Byte space and is summarized in Figure 16:

- Set FPS and map Hardware byte (FCON = 0x0C)
- Save then disable the interrupts.
- Load DPTR at address 0000h.
- Load Accumulator register with the data to load.
- Execute the MOVX @DPTR, A instruction.
- Launch the programming by writing the data sequence 54h followed by A4h in FCON register. This step of the procedure must be executed from FM1. The end of the programming indicated by the FBusy flag cleared.
- Restore the interrupts

Figure 16. Hardware Programming Procedure



Reading the Flash Spaces

User

The following procedure is used to read the User space:

- Read one byte in Accumulator by executing MOVX A, @A+DPTR with A+DPTR is the address of the code byte to read.

Note: FCON must be cleared (00h) when not used.

Extra Row

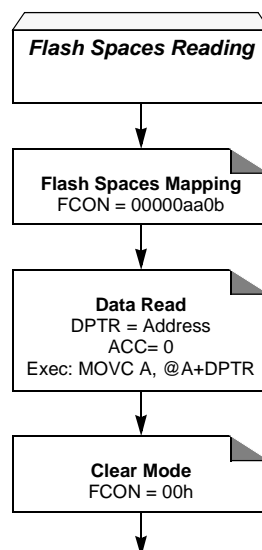
The following procedure is used to read the Extra Row space and is summarized in Figure 17:

- Map the Extra Row space by writing 02h in FCON register.
- Read one byte in Accumulator by executing MOVX A, @A+DPTR with A= 0 & DPTR= FF80h to FFFFh.
- Clear FCON to unmap the Extra Row.

Hardware Security Byte

The following procedure is used to read the Hardware Security Byte and is summarized in Figure 17:

- Map the Hardware Security space by writing 04h in FCON register.
- Read the byte in Accumulator by executing MOVX A, @A+DPTR with A= 0 & DPTR= 0000h.
- Clear FCON to unmap the Hardware Security Byte.

Figure 17. Reading Procedure

Note: aa = 10 for the Hardware Security Byte.

Flash Protection from Parallel Programming

The three lock bits in Hardware Security Byte (See 'In-System Programming' section) are programmed according to Table 24 provide different level of protection for the on-chip code and data located in FM0 and FM1.

The only way to write this bits are the parallel mode. They are set by default to level 3.

Table 24. Program Lock bit

| Program Lock bits | | | | Protection Description |
|-------------------|-----|-----|-----|---|
| Security Level | LB0 | LB1 | LB2 | |
| 1 | U | U | U | No program lock features enabled. |
| 2 | P | U | U | Parallel programming of the Flash is disabled. |
| 3 | U | P | U | Same as 2, also verify through parallel programming interface is disabled. This is the factory default programming. |
| 4 | U | U | P | Same as 3 |

Note: 1. Program Lock bits
U: unprogrammed
P: programmed

WARNING: Security level 2, 3 and 4 should only be programmed after Flash and Core verification.

Preventing Flash Corruption

See Section "Power Management".

Registers

Table 25. FCON Register
FCON Register FCON (S:D1h)
Flash Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|------|-----|-------|-------|-------|
| FPL3 | FPL2 | FPL1 | FPL0 | FPS | FMOD1 | FMOD0 | FBUSY |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 4 | FPL3:0 | Programming Launch Command bits Write 5Xh followed by AXh to launch the programming according to FMOD1:0. (See Table 23.) | | | | | |
| 3 | FPS | Flash Map Program Space Set to map the column latch space in the data memory space. Clear to re-map the data memory space. | | | | | |
| 2 - 1 | FMOD1:0 | Flash Mode See Table 22 or Table 23. | | | | | |
| 0 | FBUSY | Flash Busy Set by hardware when programming is in progress. Clear by hardware when programming is done. Can not be changed by software. | | | | | |

Reset Value = 0000 0000b

Operation Cross Memory Access

Space addressable in read and write are:

- RAM
- ERAM (Expanded RAM access by movx)
- EEPROM DATA
- FM0 (user flash)
- Hardware byte
- XROW
- Boot Flash
- Flash Column latch

The table below provides the different kind of memory which can be accessed from different code location.

Table 26. Cross Memory Access

| | Action | RAM | ERAM | Boot FLASH | FM0 | E ² Data | Hardware Byte | XROW |
|------------|--------|-----|------|------------|-------------------|---------------------|-------------------|-------------------|
| boot FLASH | Read | | | OK | OK | OK | OK | - |
| | Write | | | - | OK ⁽¹⁾ | OK ⁽¹⁾ | OK ⁽¹⁾ | OK ⁽¹⁾ |
| FM0 | Read | | | OK | OK | OK | -OK | - |
| | Write | | | - | OK (idle) | OK ⁽¹⁾ | - | -OK |

Note: 1. RWW: Read While Write

Sharing Instructions

Table 27. Instructions shared

| Action | RAM | ERAM | EEPROM DATA | Boot FLASH | FM0 | Hardware Byte | XROW |
|--------|-----|------|-------------|------------|-------|---------------|-------|
| Read | MOV | MOVX | MOVX | MOVC | MOVC | MOVC | MOVC |
| Write | MOV | MOVX | MOVX | - | by cl | by cl | by cl |

Note: by cl : using Column Latch

Table 28. Read MOVX A, @DPTR

| EEE bit in EECON Register | FPS in FCON Register | ENBOOT | ERAM | EEPROM DATA | Flash Column Latch |
|---------------------------|----------------------|--------|------|-------------|--------------------|
| 0 | 0 | X | OK | | |
| 0 | 1 | X | OK | | |
| 1 | 0 | X | | OK | |
| 1 | 1 | X | OK | | |

Table 29. Write MOVX @DPTR,A

| EEE bit in EECON Register | FPS bit in FCON Register | ENBOOT | ERAM | EEPROM Data | Flash Column Latch |
|---------------------------|--------------------------|--------|------|-------------|--------------------|
| 0 | 0 | X | OK | | |
| 0 | 1 | X | | | OK |
| 1 | 0 | X | | OK | |
| 1 | 1 | X | | | OK |

Table 30. Read MOV C A, @DPTR

| Code Execution | FCON Register | | | ENBOOT | DPTR | FM1 | FM0 | XROW | Hardware Byte |
|----------------|---------------|-------|-----|--------|-------------------------------------|-------------------------------|-----|------|---------------|
| | FMOD1 | FMOD0 | FPS | | | | | | |
| From FM0 | 0 | 0 | X | 0 | 0000h to 3FFFh | | OK | | |
| | | | | 1 | 0000h to 3FFFh | | OK | | |
| | | | | | F800h to FFFFh | Do not use this configuration | | | |
| | 0 | 1 | X | X | 0000 to 007Fh See ⁽¹⁾ | | | OK | |
| | 1 | 0 | X | X | X | | | | OK |
| | 1 | 1 | X | 0 | 000h to 3FFFh | | OK | | |
| | | | | 1 | 0000h to 3FFFh | | OK | | |
| | | | | | F800h to FFFFh | Do not use this configuration | | | |
| | 0 | 0 | 0 | 1 | 0000h to 3FFF | | OK | | |
| | | | | | F800h to FFFFh | OK | | | |
| | | | | 0 | X | NA | | | |
| | | | 1 | 1 | X | | OK | | |
| | | | | 0 | X | NA | | | |
| | | | X | 1 | 0000h to 007h See ⁽²⁾ | | | OK | |
| | | | | 0 | | NA | | | |
| | | | X | 1 | X | | | | OK |
| | | | | 0 | | NA | | | |
| | 1 | 1 | X | 1 | 000h to 3FFFh | | OK | | |
| | | | | 0 | | NA | | | |

1. For DPTR higher than 007Fh only lowest 7 bits are decoded, thus the behavior is the same as for addresses from 0000h to 007Fh
2. For DPTR higher than 007Fh only lowest 7 bits are decoded, thus the behavior is the same as for addresses from 0000h to 007Fh

In-System Programming (ISP)

With the implementation of the User Space (FM0) and the Boot Space (FM1) in Flash technology the T89C51CC02 allows the system engineer the development of applications with a very high level of flexibility. This flexibility is based on the possibility to alter the customer program at any stages of a product's life:

- Before mounting the chip on the PCB, FM0 flash can be programmed with the application code. FM1 is always preprogrammed by Atmel with a bootloader (chip can be ordered with CAN bootloader or UART bootloader).⁽¹⁾
- Once the chip is mounted on the PCB, it can be programmed by serial mode via the CAN bus or UART.

Note: 1. The user can also program his own bootloader in FM1.

This ISP allows code modification over the total lifetime of the product.

Besides the default Bootloaders Atmel provide customers all the needed Application-Programming-Interfaces (API) which are needed for the ISP. The API are located in the Boot memory.

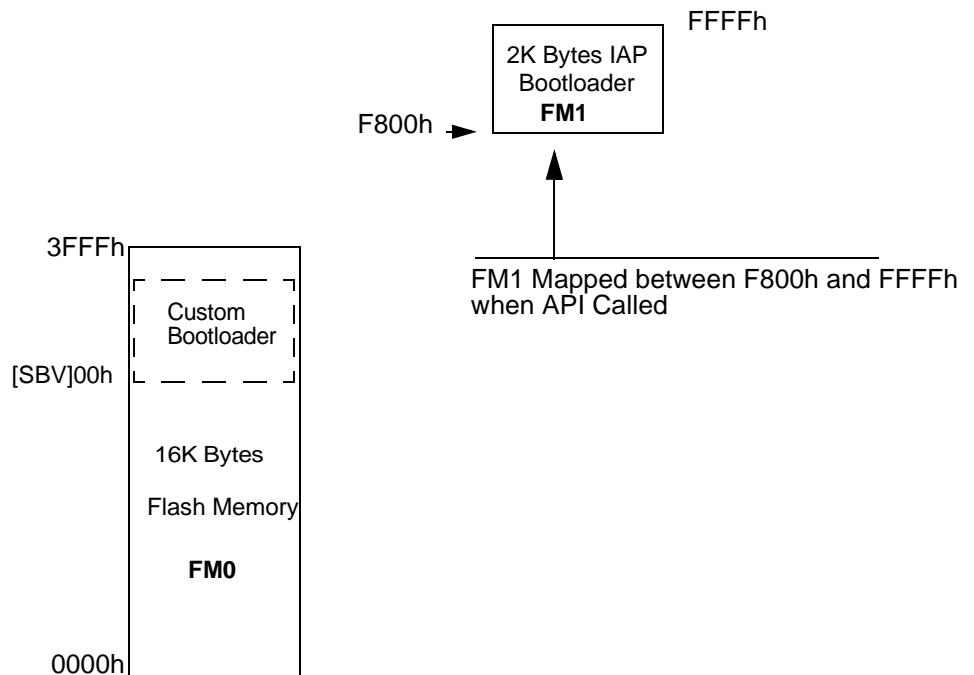
This allow the customer to have a full use of the 16-Kbyte user memory.

Flash Programming and Erasure

There are three methods for programming the Flash memory:

- The Atmel bootloader located in FM1 is activated by the application. Low level API routines (located in FM1) will be used to program FM0. The interface used for serial downloading to FM0 is the UART or the CAN. API can be called also by user's bootloader located in FM0 at [SBV]00h.
- A further method exist in activating the Atmel boot loader by hardware activation. See the Section "Hardware Security Byte".
- The FM0 can be programmed also by the parallel mode using a programmer.

Figure 18. Flash Memory Mapping



Boot Process

Software Boot Process Example

Many algorithms can be used for the software boot process. Below are descriptions of the different flags and Bytes.

Boot Loader Jump bit (BLJB):

- This bit indicates if on RESET the user wants to jump to this application at address @0000h on FM0 or execute the boot loader at address @F800h on FM1.
- BLJB = 0 (i.e. bootloader FM1 executed after a reset) is the default Atmel factory programming.
- To read or modify this bit, the APIs are used.

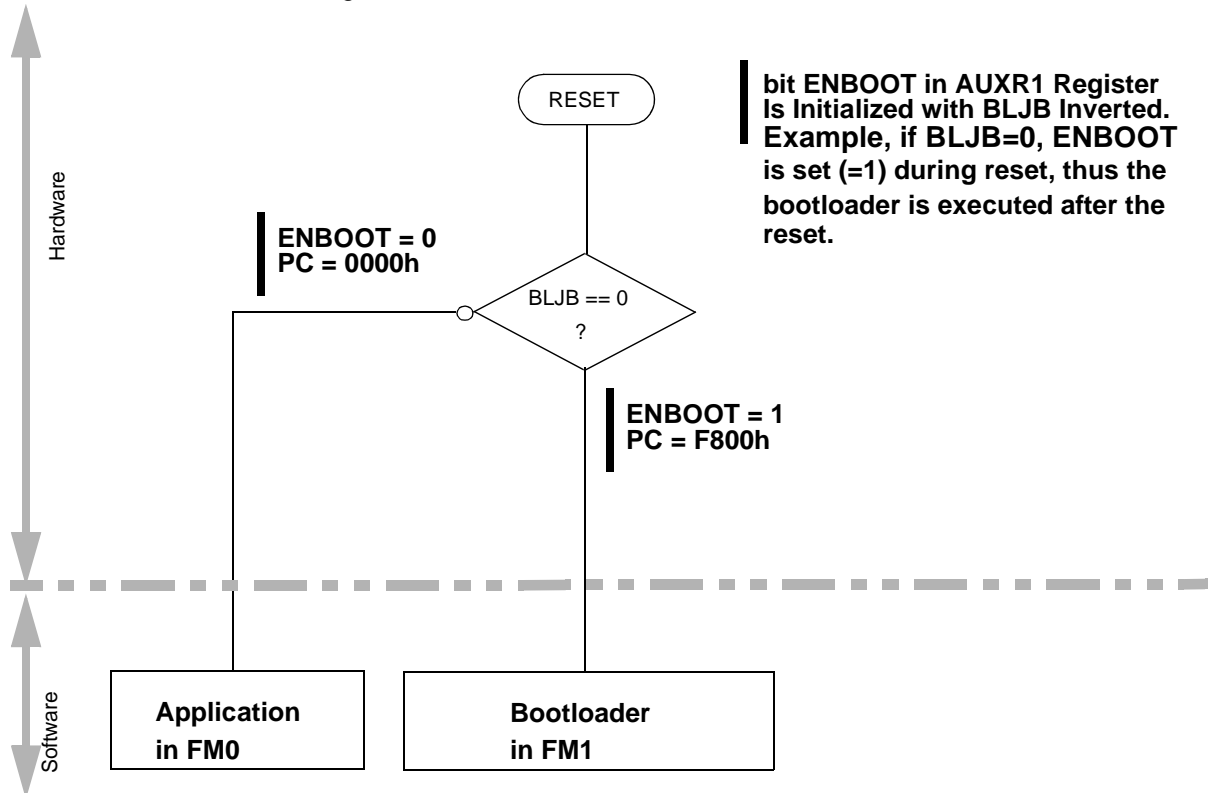
Boot Vector Address (SBV):

- This byte contains the MSB of the user boot loader address in FM0.
- The default value of SBV is FCh (no user boot loader in FM0).
- To read or modify this byte, the APIs are used.

Extra Byte (EB) & Boot Status Byte (BSB):

- These Bytes are reserved for customer use.
- To read or modify these Bytes, the APIs are used.

Figure 19. Hardware Boot Process Algorithm



Application-Programming-Interface

Several Application Program Interface (API) calls are available for use by an application program to permit selective erasing and programming of Flash pages. All calls are made by functions.

All these APIs are described in detail in the following documents on the Atmel web site.

- Datasheet Bootloader CAN T89C51CC02.
- Datasheet Bootloader UART T89C51CC02.

XROW Bytes

The EXTRA ROW (XROW) includes 128 bytes. Some of these bytes are used for specific purpose in conjunction with the bootloader.

Table 31. XROW Mapping

| Description | Default Value | Address |
|---|---------------|---------|
| Copy of the Manufacturer Code | 58h | 30h |
| Copy of the Device ID#1: Family code | D7h | 31h |
| Copy of the Device ID#2: Memories size and type | BBh | 60h |
| Copy of the Device ID#3: Name and Revision | FFh | 61h |

Hardware Conditions

It is possible to force the controller to execute the bootloader after a Reset with hardware conditions.

During the first programming, the user can define a configuration on Port1 that will be recognized by the chip as the hardware conditions during a Reset. If this condition is met, the chip will start executing the bootloader at the end of the Reset.

See a detailed description in the applicable Document.

- Datasheet Bootloader CAN T89C51CC02.
- Datasheet Bootloader UART T89C51CC02.

Hardware Security Byte

Table 32. Hardware Security byte

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|------------|------------|------------|
| X2B | BLJB | - | - | - | LB2 | LB1 | LB0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | X2B | X2 bit Set this bit to start in standard mode Clear this bit to start in X2 Mode. | | | | | |
| 6 | BLJB | Boot Loader Jump bit - 1: To start the user's application on next RESET (@0000h) located in FM0, - 0: To start the boot loader(@F800h) located in FM1. | | | | | |
| 5 - 3 | - | Reserved The value read from these bits are indeterminate. | | | | | |
| 2 - 0 | LB2:0 | Lock bits (see Table 22) | | | | | |

After erasing the chip in parallel mode, the default value is : FFh

The erasing in ISP mode (from bootloader) does not modify this byte.

- Notes:
1. Only the 4 MSB bits can be accessed by software.
 2. The 4 LSB bits can only be accessed by parallel mode.

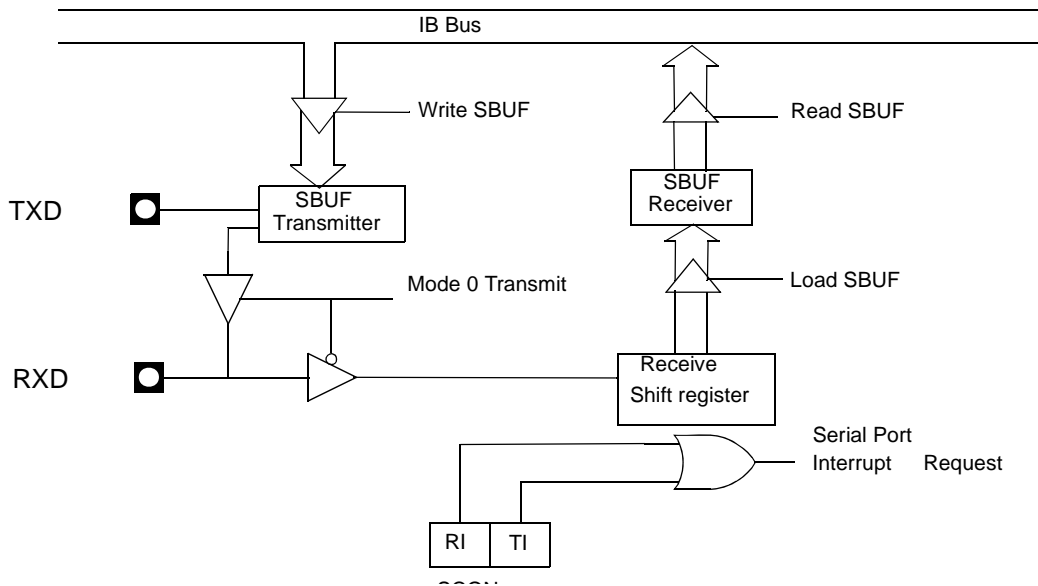
Serial I/O Port

The T89C51CC02 I/O serial port is compatible with the I/O serial port in the 80C52. It provides both synchronous and asynchronous communication modes. It operates as a Universal Asynchronous Receiver and Transmitter (UART) in three full-duplex modes (Modes 1, 2 and 3). Asynchronous transmission and reception can occur simultaneously and at different baud rates

Serial I/O port includes the following enhancements:

- Framing error detection
- Automatic address recognition

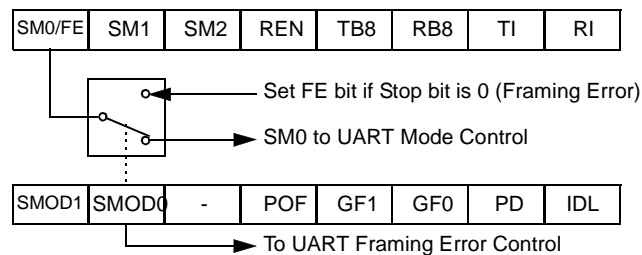
Figure 20. Serial I/O Port Block Diagram



Framing Error Detection

Framing bit error detection is provided for the three asynchronous modes. To enable the framing bit error detection feature, set SMOD0 bit in PCON register.

Figure 21. Framing Error Block Diagram



When this feature is enabled, the receiver checks each incoming data frame for a valid stop bit. An invalid stop bit may result from noise on the serial lines or from simultaneous transmission by two CPUs. If a valid stop bit is not found, the Framing Error bit (FE) in SCON register bit is set.

The software may examine the FE bit after each reception to check for data errors. Once set, only software or a reset clears the FE bit. Subsequently received frames with valid stop bits cannot clear the FE bit. When the FE feature is enabled, RI rises on the stop bit instead of the last data bit (See Figure 22 and Figure 23).

Figure 22. UART Timing in Mode 1

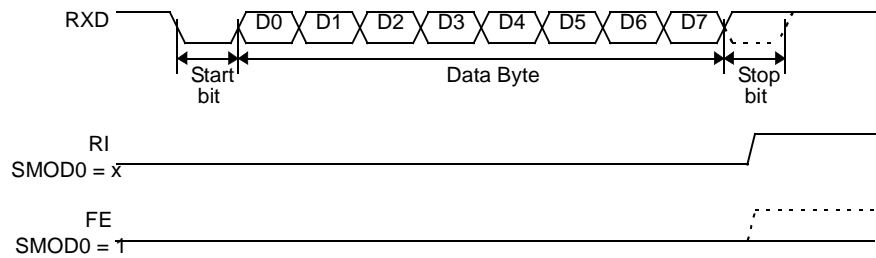
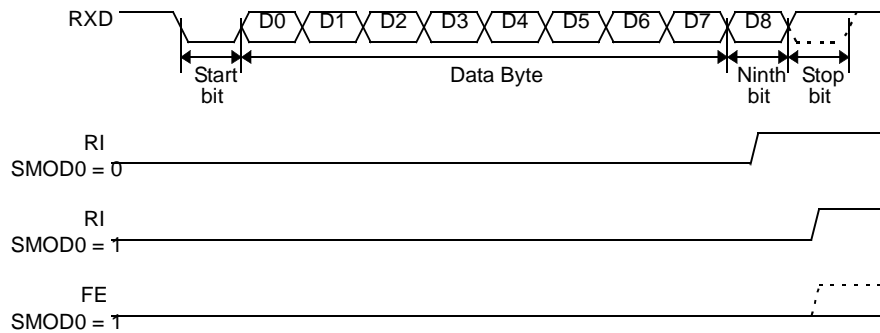


Figure 23. UART Timing in Modes 2 and 3



Automatic Address Recognition

The automatic address recognition feature is enabled when the multiprocessor communication feature is enabled (SM2 bit in SCON register is set).

Implemented in the hardware, automatic address recognition enhances the multiprocessor communication feature by allowing the serial port to examine the address of each incoming command frame. Only when the serial port recognizes its own address will the receiver set the RI bit in the SCON register to generate an interrupt. This ensures that the CPU is not interrupted by command frames addressed to other devices.

If necessary, the user can enable the automatic address recognition feature in mode 1. In this configuration, the stop bit takes the place of the ninth data bit. bit RI is set only when the received command frame address matches the device's address and is terminated by a valid stop bit.

To support automatic address recognition, a device is identified by a given address and a broadcast address.

Note: The multiprocessor communication and automatic address recognition features cannot be enabled in mode 0 (i.e. setting SM2 bit in SCON register in mode 0 has no effect).

Given Address

Each device has an individual address that is specified in the SADDR register; the SADEN register is a mask byte that contains don't-care bits (defined by zeros) to form the device's given address. The don't-care bits provide the flexibility to address one or more slaves at a time. The following example illustrates how a given address is formed. To address a device by its individual address, the SADEN mask byte must be 1111 1111b.

For example:

```
SADDR0101 0110b
SADEN1111 1100b
Given0101 01XXb
```

Here is an example of how to use given addresses to address different slaves:

Slave A:SADDR1111 0001b
SADEN1111 1010b
 Given1111 0X0Xb

Slave B:SADDR1111 0011b
SADEN1111 1001b
 Given1111 0XX1b

Slave C:SADDR1111 0011b
SADEN1111 1101b
 Given1111 00X1b

The SADEN byte is selected so that each slave may be addressed separately.

For slave A, bit 0 (the LSB) is a don't-care bit; for slaves B and C, bit 0 is a 1. To communicate with slave A only, the master must send an address where bit 0 is clear (e.g. 1111 0000b).

For slave A, bit 1 is a 0; for slaves B and C, bit 1 is a don't care bit. To communicate with slaves A and B, but not slave C, the master must send an address with bits 0 and 1 both set (e.g. 1111 0011b).

To communicate with slaves A, B and C, the master must send an address with bit 0 set, bit 1 clear, and bit 2 clear (e.g. 1111 0001b).

Broadcast Address

A broadcast address is formed from the logical OR of the SADDR and SADEN registers with zeros defined as don't-care bits, e.g.:

SADDR 0101 0110b
SADEN 1111 1100b
 SADDR OR SADEN1111 111Xb

The use of don't-care bits provides flexibility in defining the broadcast address, however in most applications, a broadcast address is FFh. The following is an example of using broadcast addresses:

Slave A:SADDR1111 0001b
SADEN1111 1010b
 Given1111 1X11b,

Slave B:SADDR1111 0011b
SADEN1111 1001b
 Given1111 1X11B,

Slave C:SADDR=1111 0010b
SADEN1111 1101b
 Given1111 1111b

For slaves A and B, bit 2 is a don't care bit; for slave C, bit 2 is set. To communicate with all of the slaves, the master must send an address FFh. To communicate with slaves A and B, but not slave C, the master can send an address FBh.

Registers

Table 33. SCON Register

SCON (S:98h)
Serial Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | |
|------------|--------------|--|--|-----|-----|----|----|-----|-----|------|-----------|---|---|----------------|--|---|---|------------|----------|---|---|-----------|--------------------------------|---|---|-----------|----------|
| FE/SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI | | | | | | | | | | | | | | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | FE | Framing Error bit (SMOD0 = 1) Clear to reset the error state, not cleared by a valid stop bit. Set by hardware when an invalid stop bit is detected. | | | | | | | | | | | | | | | | | | | | | | | | | |
| | SM0 | Serial port Mode bit 0 (SMOD0 = 0) Refer to SM1 for serial port mode selection. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | SM1 | Serial port Mode bit 1 <table><thead><tr><th>SM0</th><th>SM1</th><th>Mode</th><th>Baud Rate</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>Shift Register</td><td>$F_{XTAL}/12$ (or $F_{XTAL}/6$ in mode X2)</td></tr><tr><td>0</td><td>1</td><td>8-bit UART</td><td>Variable</td></tr><tr><td>1</td><td>0</td><td>9bit UART</td><td>$F_{XTAL}/64$ or $F_{XTAL}/32$</td></tr><tr><td>1</td><td>1</td><td>9bit UART</td><td>Variable</td></tr></tbody></table> | | | | | | SM0 | SM1 | Mode | Baud Rate | 0 | 0 | Shift Register | $F_{XTAL}/12$ (or $F_{XTAL}/6$ in mode X2) | 0 | 1 | 8-bit UART | Variable | 1 | 0 | 9bit UART | $F_{XTAL}/64$ or $F_{XTAL}/32$ | 1 | 1 | 9bit UART | Variable |
| SM0 | SM1 | Mode | Baud Rate | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | Shift Register | $F_{XTAL}/12$ (or $F_{XTAL}/6$ in mode X2) | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 8-bit UART | Variable | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 9bit UART | $F_{XTAL}/64$ or $F_{XTAL}/32$ | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 9bit UART | Variable | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | SM2 | Serial port Mode 2 bit/Multiprocessor Communication Enable bit Clear to disable multiprocessor communication feature. Set to enable multiprocessor communication feature in mode 2 and 3. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | REN | Reception Enable bit Clear to disable serial reception. Set to enable serial reception. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | TB8 | Transmitter bit 8/Ninth bit to Transmit in Modes 2 and 3 Clear to transmit a logic 0 in the 9th bit. Set to transmit a logic 1 in the 9th bit. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | RB8 | Receiver bit 8/Ninth bit Received in Modes 2 and 3 Cleared by hardware if 9th bit received is a logic 0. Set by hardware if 9th bit received is a logic 1. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | TI | Transmit Interrupt Flag Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0 or at the beginning of the stop bit in the other modes. | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | RI | Receive Interrupt Flag Clear to acknowledge interrupt. Set by hardware at the end of the 8th bit time in mode 0, See Figure 22. and Figure 23. in the other modes. | | | | | | | | | | | | | | | | | | | | | | | | | |

Reset Value = 0000 0000b
bit addressable

Table 34. SADEN Register
 SADEN (S:B9h)
 Slave Address Mask Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | Mask Data for Slave Individual Address | | | | | |

Reset Value = 0000 0000b
 Not bit addressable

Table 35. SADDR Register
 SADDR (S:A9h)
 Slave Address Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--------------------------|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | Slave Individual Address | | | | | |

Reset Value = 0000 0000b
 Not bit addressable

Table 36. SBUF Register
 SBUF (S:99h)
 Serial Data Buffer

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---------------------------------------|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | Data sent/received by Serial I/O Port | | | | | |

Reset Value = 0000 0000b
 Not bit addressable

Table 37. PCON Register
PCON (S:87h)
Power Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-----|-----|-----|----|-----|
| SMOD1 | SMOD0 | - | POF | GF1 | GF0 | PD | IDL |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | SMOD1 | Serial port Mode bit 1 Set to select double baud rate in mode 1, 2 or 3. | | | | | |
| 6 | SMOD0 | Serial port Mode bit 0 Clear to select SM0 bit in SCON register. Set to select FE bit in SCON register. | | | | | |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 4 | POF | Power-off Flag Clear to recognize next reset type. Set by hardware when V_{CC} rises from 0 to its nominal voltage. Can also be set by software. | | | | | |
| 3 | GF1 | General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage. | | | | | |
| 2 | GF0 | General purpose Flag Cleared by user for general purpose usage. Set by user for general purpose usage. | | | | | |
| 1 | PD | Power-down Mode bit Cleared by hardware when reset occurs. Set to enter power-down mode. | | | | | |
| 0 | IDL | Idle Mode bit Clear by hardware when interrupt or reset occurs. Set to enter idle mode. | | | | | |

Reset Value = 00X1 0000b
Not bit addressable

Timers/Counters

The T89C51CC02 implements two general-purpose, 16-bit Timers/Counters. Such are identified as Timer 0 and Timer 1, and can be independently configured to operate in a variety of modes as a Timer or an event Counter. When operating as a Timer, the Timer/Counter runs for a programmed length of time, then issues an interrupt request. When operating as a Counter, the Timer/Counter counts negative transitions on an external pin. After a preset number of counts, the Counter issues an interrupt request. The various operating modes of each Timer/Counter are described in the following sections.

Timer/Counter Operations

A basic operation is Timer registers THx and TLx ($x = 0, 1$) connected in cascade to form a 16-bit Timer. Setting the run control bit (TRx) in TCON register (See Figure 38) turns the Timer on by allowing the selected input to increment TLx. When TLx overflows it increments THx; when THx overflows it sets the Timer overflow flag (TFx) in TCON register. Setting the TRx does not clear the THx and TLx Timer registers. Timer registers can be accessed to obtain the current count or to enter preset values. They can be read at any time but TRx bit must be cleared to preset their values, otherwise the behavior of the Timer/Counter is unpredictable.

The C/Tx# control bit selects Timer operation or Counter operation by selecting the divided-down peripheral clock or external pin Tx as the source for the counted signal. TRx bit must be cleared when changing the mode of operation, otherwise the behavior of the Timer/Counter is unpredictable.

For Timer operation ($C/Tx\# = 0$), the Timer register counts the divided-down peripheral clock. The Timer register is incremented once every peripheral cycle (6 peripheral clock periods). The Timer clock rate is $f_{PER}/6$, i.e. $f_{OSC}/12$ in standard mode or $f_{OSC}/6$ in X2 Mode.

For Counter operation ($C/Tx\# = 1$), the Timer register counts the negative transitions on the Tx external input pin. The external input is sampled every peripheral cycles. When the sample is high in one cycle and low in the next one, the Counter is incremented. Since it takes 2 cycles (12 peripheral clock periods) to recognize a negative transition, the maximum count rate is $f_{PER}/12$, i.e. $f_{OSC}/24$ in standard mode or $f_{OSC}/12$ in X2 Mode. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full peripheral cycle.

Timer 0

Timer 0 functions as either a Timer or event Counter in four modes of operation. Figure 24 through Figure 27 show the logical configuration of each mode.

Timer 0 is controlled by the four lower bits of TMOD register (See Figure 39) and bits 0, 1, 4 and 5 of TCON register (See Figure 38). TMOD register selects the method of Timer gating (GATE0), Timer or Counter operation (T/C0#) and mode of operation (M10 and M00). TCON register provides Timer 0 control functions: overflow flag (TF0), run control bit (TR0), interrupt flag (IE0) and interrupt type control bit (IT0).

For normal Timer operation ($GATE0 = 0$), setting TR0 allows TL0 to be incremented by the selected input. Setting GATE0 and TR0 allows external pin INT0# to control Timer operation.

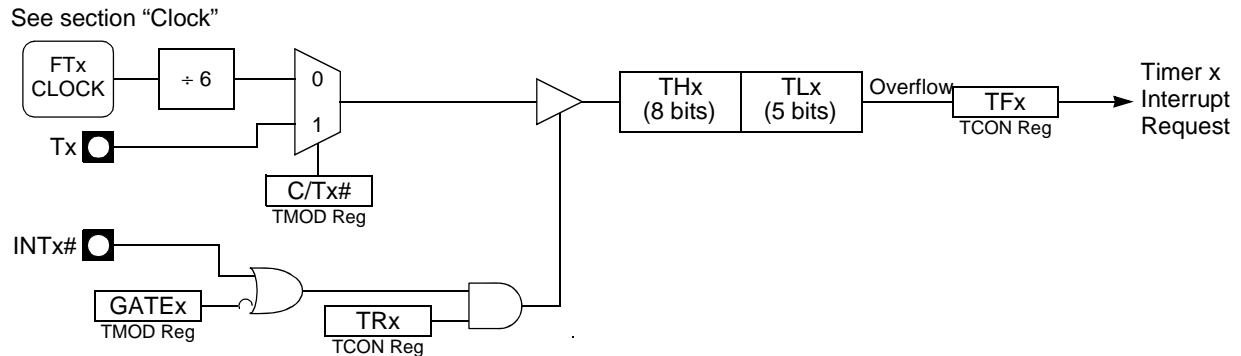
Timer 0 overflow (count rolls over from all 1s to all 0s) sets TF0 flag generating an interrupt request.

It is important to stop Timer/Counter before changing mode.

Mode 0 (13-bit Timer)

Mode 0 configures Timer 0 as an 13-bit Timer which is set up as an 8-bit Timer (TH0 register) with a modulo 32 prescaler implemented with the lower five bits of TL0 register (See Figure 24). The upper three bits of TL0 register are indeterminate and should be ignored. Prescaler overflow increments TH0 register.

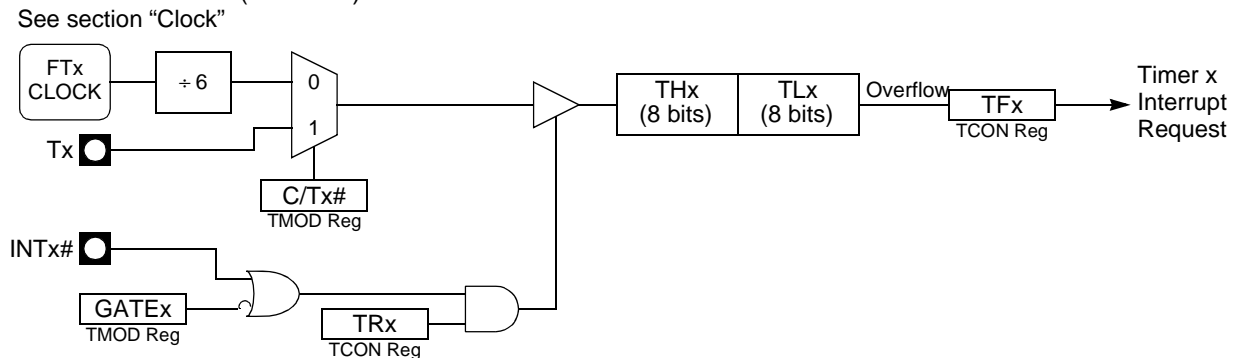
Figure 24. Timer/Counter x (x= 0 or 1) in Mode 0



Mode 1 (16-bit Timer)

Mode 1 configures Timer 0 as a 16-bit Timer with TH0 and TL0 registers connected in cascade (See Figure 25). The selected input increments TL0 register.

Figure 25. Timer/Counter x (x= 0 or 1) in Mode 1

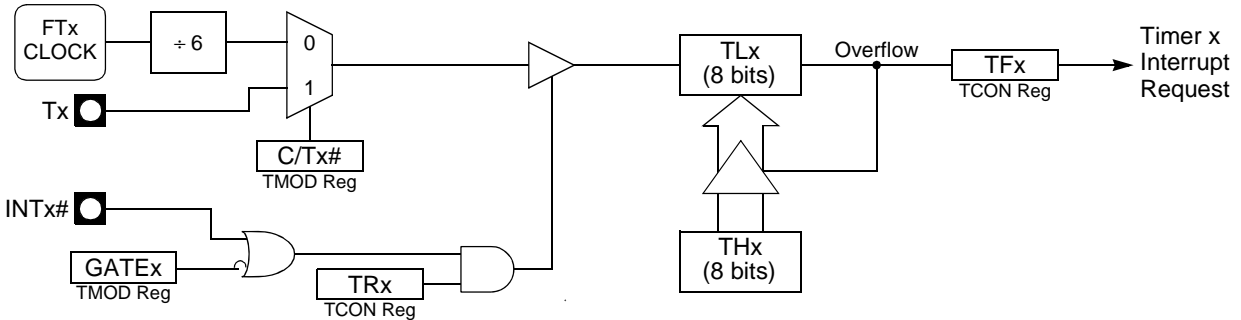


Mode 2 (8-bit Timer with Auto-Reload)

Mode 2 configures Timer 0 as an 8-bit Timer (TL0 register) that automatically reloads from TH0 register (See Figure 26). TL0 overflow sets TF0 flag in TCON register and reloads TL0 with the contents of TH0, which is preset by software. When the interrupt request is serviced, hardware clears TF0. The reload leaves TH0 unchanged. The next reload value may be changed at any time by writing it to TH0 register.

Figure 26. Timer/Counter x (x= 0 or 1) in Mode 2

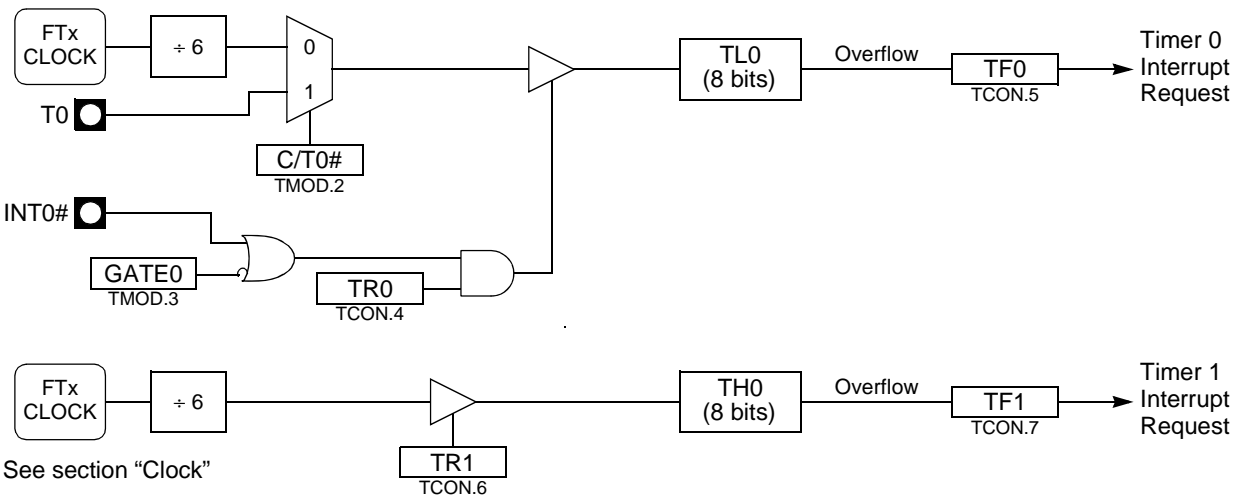
See section "Clock"



Mode 3 (Two 8-bit Timers)

Mode 3 configures Timer 0 such that registers TL0 and TH0 operate as separate 8-bit Timers (See Figure 27). This mode is provided for applications requiring an additional 8-bit Timer or Counter. TL0 uses the Timer 0 control bits C/T0# and GATE0 in TMOD register, and TR0 and TF0 in TCON register in the normal manner. TH0 is locked into a Timer function (counting $F_{PER}/6$) and takes over use of the Timer 1 interrupt (TF1) and run control (TR1) bits. Thus, operation of Timer 1 is restricted when Timer 0 is in mode 3.

Figure 27. Timer/Counter 0 in Mode 3: Two 8-bit Counters



Timer 1

Timer 1 is identical to Timer 0 excepted for Mode 3 which is a hold-count mode. Following comments help to understand the differences:

- Timer 1 functions as either a Timer or event Counter in three modes of operation. Figure 24 to Figure 26 show the logical configuration for modes 0, 1, and 2. Timer 1's mode 3 is a hold-count mode.
- Timer 1 is controlled by the four high-order bits of TMOD register (See Figure 39) and bits 2, 3, 6 and 7 of TCON register (See Figure 38). TMOD register selects the method of Timer gating (GATE1), Timer or Counter operation (C/T1#) and mode of operation (M11 and M01). TCON register provides Timer 1 control functions: overflow flag (TF1), run control bit (TR1), interrupt flag (IE1) and interrupt type control bit (IT1).
- Timer 1 can serve as the Baud Rate Generator for the Serial Port. Mode 2 is best suited for this purpose.

- For normal Timer operation (GATE1= 0), setting TR1 allows TL1 to be incremented by the selected input. Setting GATE1 and TR1 allows external pin INT1# to control Timer operation.
- Timer 1 overflow (count rolls over from all 1s to all 0s) sets the TF1 flag generating an interrupt request.
- When Timer 0 is in mode 3, it uses Timer 1's overflow flag (TF1) and run control bit (TR1). For this situation, use Timer 1 only for applications that do not require an interrupt (such as a Baud Rate Generator for the Serial Port) and switch Timer 1 in and out of mode 3 to turn it off and on.
- It is important to stop Timer/Counter before changing mode.

Mode 0 (13-bit Timer)

Mode 0 configures Timer 1 as a 13-bit Timer, which is set up as an 8-bit Timer (TH1 register) with a modulo-32 prescaler implemented with the lower 5 bits of the TL1 register (See Figure 24). The upper 3 bits of TL1 register are ignored. Prescaler overflow increments TH1 register.

Mode 1 (16-bit Timer)

Mode 1 configures Timer 1 as a 16-bit Timer with TH1 and TL1 registers connected in cascade (See Figure 25). The selected input increments TL1 register.

Mode 2 (8-bit Timer with Auto-Reload)

Mode 2 configures Timer 1 as an 8-bit Timer (TL1 register) with automatic reload from TH1 register on overflow (See Figure 26). TL1 overflow sets TF1 flag in TCON register and reloads TL1 with the contents of TH1, which is preset by software. The reload leaves TH1 unchanged.

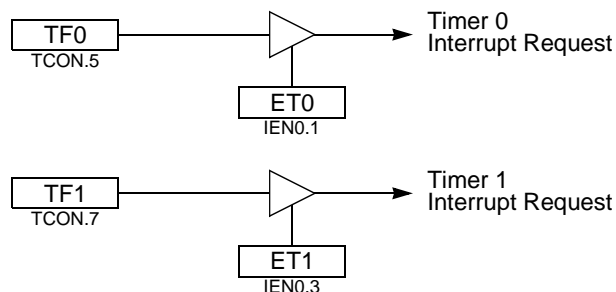
Mode 3 (Halt)

Placing Timer 1 in mode 3 causes it to halt and hold its count. This can be used to halt Timer 1 when TR1 run control bit is not available i.e. when Timer 0 is in mode 3.

Interrupt

Each Timer handles one interrupt source that is the timer overflow flag TF0 or TF1. This flag is set every time an overflow occurs. Flags are cleared when vectoring to the Timer interrupt routine. Interrupts are enabled by setting ETx bit in IEN0 register. This assumes interrupts are globally enabled by setting EA bit in IEN0 register.

Figure 28. Timer Interrupt System



Registers

Table 38. TCON Register
TCON (S:88h)
Timer/Counter Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-----|-----|-----|-----|-----|
| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | TF1 | Timer 1 Overflow Flag Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 1 register overflows. | | | | | |
| 6 | TR1 | Timer 1 Run Control bit Clear to turn off Timer/Counter 1. Set to turn on Timer/Counter 1. | | | | | |
| 5 | TF0 | Timer 0 Overflow Flag Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 0 register overflows. | | | | | |
| 4 | TR0 | Timer 0 Run Control bit Clear to turn off Timer/Counter 0. Set to turn on Timer/Counter 0. | | | | | |
| 3 | IE1 | Interrupt 1 Edge Flag Cleared by hardware when interrupt is processed if edge-triggered (See IT1). Set by hardware when external interrupt is detected on INT1# pin. | | | | | |
| 2 | IT1 | Interrupt 1 Type Control bit Clear to select low level active (level triggered) for external interrupt 1 (INT1#). Set to select falling edge active (edge triggered) for external interrupt 1. | | | | | |
| 1 | IE0 | Interrupt 0 Edge Flag Cleared by hardware when interrupt is processed if edge-triggered (See IT0). Set by hardware when external interrupt is detected on INT0# pin. | | | | | |
| 0 | IT0 | Interrupt 0 Type Control bit Clear to select low level active (level triggered) for external interrupt 0 (INT0#). Set to select falling edge active (edge triggered) for external interrupt 0. | | | | | |

Reset Value = 0000 0000b

Table 39. TMOD Register
TMOD (S:89h)
Timer/Counter Mode Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|-----|-----|-------|-------|-----|-----|
| GATE1 | C/T1# | M11 | M01 | GATE0 | C/T0# | M10 | M00 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | GATE1 | Timer 1 Gating Control bit Clear to enable Timer 1 whenever TR1 bit is set. Set to enable Timer 1 only while INT1# pin is high and TR1 bit is set. |
| 6 | C/T1# | Timer 1 Counter/Timer Select bit Clear for Timer operation: Timer 1 counts the divided-down system clock. Set for Counter operation: Timer 1 counts negative transitions on external pin T1. |
| 5 | M11 | Timer 1 Mode Select bits <u>M11 M01 Operating mode</u> 0 0 Mode 0: 8-bit Timer/Counter (TH1) with 5bit prescaler (TL1). 0 1 Mode 1: 16-bit Timer/Counter. 1 1 Mode 3: Timer 1 halted. Retains count. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL1). ⁽¹⁾ |
| 4 | M01 | |
| 3 | GATE0 | |
| 2 | C/T0# | |
| 1 | M10 | Timer 0 Mode Select bit <u>M10 M00 Operating mode</u> 0 0 Mode 0: 8-bit Timer/Counter (TH0) with 5bit prescaler (TL0). 0 1 Mode 1: 16-bit Timer/Counter. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL0). ⁽²⁾ 1 1 Mode 3: TL0 is an 8-bit Timer/Counter. TH0 is an 8-bit Timer using Timer 1's TR0 and TF0 bits. |
| 0 | M00 | |

Reset Value = 0000 0000b

Notes: 1. Reloaded from TH1 at overflow.
2. Reloaded from TH0 at overflow.

Table 40. TH0 Register
TH0 (S:8Ch)
Timer 0 High Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|-----------------------------|
| 7:0 | | High Byte of Timer 0 |

Reset Value = 0000 0000b

Table 41. TL0 Register
 TL0 (S:8Ah)
 Timer 0 Low Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---------------------|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7:0 | | Low Byte of Timer 0 | | | | | |

Reset Value = 0000 0000b

Table 42. TH1 Register
 TH1 (S:8Dh)
 Timer 1 High Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|----------------------|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7:0 | | High Byte of Timer 1 | | | | | |

Reset Value = 0000 0000b

Table 43. TL1 Register
 TL1 (S:8Bh)
 Timer 1 Low Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---------------------|---|---|---|---|---|
| | | | | | | | |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7:0 | | Low Byte of Timer 1 | | | | | |

Reset Value = 0000 0000b

Timer 2

The T89C51CC02 Timer 2 is compatible with Timer 2 in the 80C52.

It is a 16-bit timer/counter: the count is maintained by two eightbit timer registers, TH2 and TL2 that are cascade-connected. It is controlled by T2CON register (See Table 45) and T2MOD register (See Table 46). Timer 2 operation is similar to Timer 0 and Timer 1. $C/\overline{T}2$ selects $F_{T2\text{ clock}}/6$ (timer operation) or external pin T2 (counter operation) as timer clock. Setting TR2 allows TL2 to be incremented by the selected input.

Timer 2 includes the following enhancements:

- Auto-reload mode (up or down counter)
- Programmable clock-output

Auto-Reload Mode

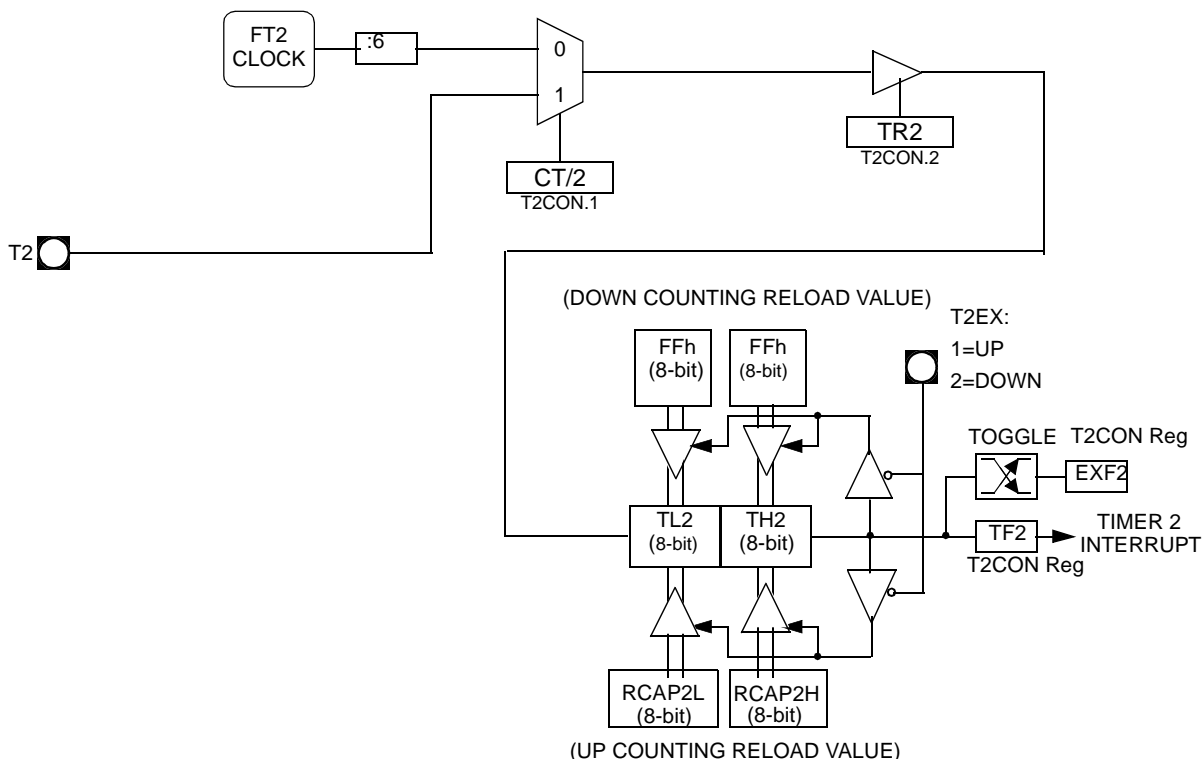
The auto-reload mode configures Timer 2 as a 16-bit timer or event counter with automatic reload. This feature is controlled by the DCEN bit in T2MOD register (See Table 45). Setting the DCEN bit enables Timer 2 to count up or down as shown in Figure 29. In this mode the T2EX pin controls the counting direction.

When T2EX is high, Timer 2 counts up. Timer overflow occurs at FFFFh which sets the TF2 flag and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L registers to be loaded into the timer registers TH2 and TL2.

When T2EX is low, Timer 2 counts down. Timer underflow occurs when the count in the timer registers TH2 and TL2 equals the value stored in RCAP2H and RCAP2L registers. The underflow sets TF2 flag and reloads FFFFh into the timer registers.

The EXF2 bit toggles when Timer 2 overflow or underflow, depending on the direction of the count. EXF2 does not generate an interrupt. This bit can be used to provide 17-bit resolution.

Figure 29. Auto-Reload Mode Up/Down Counter
See section "Clock"



Programmable Clock-Output

In clock-out mode, Timer 2 operates as a 50%-duty-cycle, programmable clock generator (Figure 30). The input clock increments TL2 at frequency $f_{OSC}/2$. The timer repeatedly counts to overflow from a loaded value. At overflow, the contents of RCAP2H and RCAP2L registers are loaded into TH2 and TL2. In this mode, Timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency depending on the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

$$Clock - OutFrequency = \frac{FT2clock}{4 \times (65536 - RCAP2H/RCAP2L)}$$

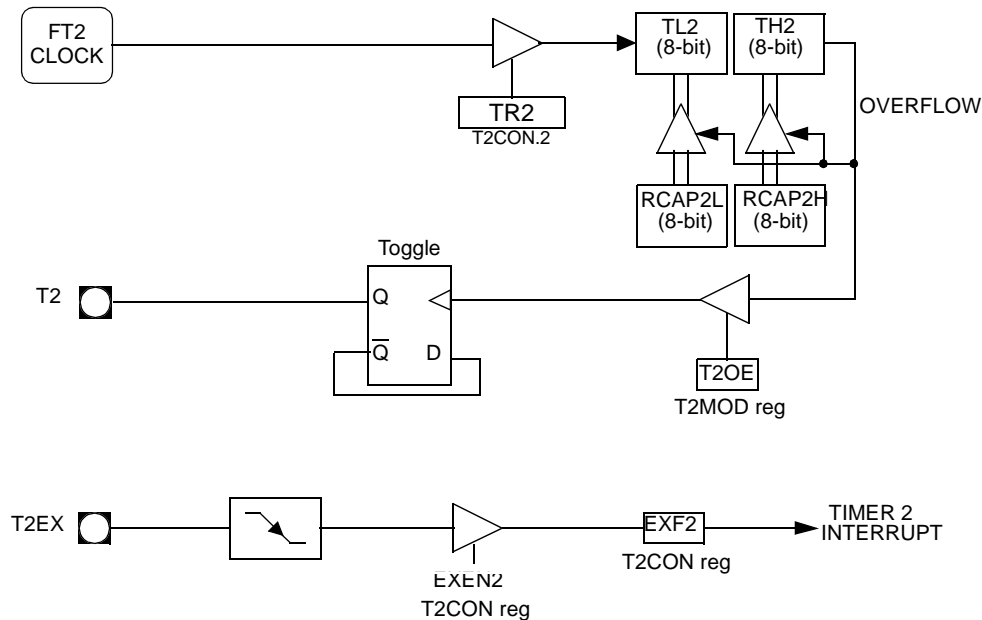
For a 16 MHz system clock in x1 mode, Timer 2 has a programmable frequency range of 61 Hz ($f_{OSC}/2^{16}$) to 4 MHz ($f_{OSC}/4$). The generated clock signal is brought out to T2 pin (P1.0).

Timer 2 is programmed for the clock-out mode as follows:

- Set T2OE bit in T2MOD register.
- Clear $\overline{C/T2}$ bit in T2CON register.
- Determine the 16-bit reload value from the formula and enter it in RCAP2H/RCAP2L registers.
- Enter a 16-bit initial value in timer registers TH2/TL2. It can be the same as the reload value or different depending on the application.
- To start the timer, set TR2 run control bit in T2CON register.

It is possible to use Timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers.

Figure 30. Clock-Out Mode



Registers

Table 44. T2CON Register
T2CON (S:C8h)
Timer 2 Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|------|-------|-----|-------|---------|
| TF2 | EXF2 | RCLK | TCLK | EXEN2 | TR2 | C/T2# | CP/RL2# |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | TF2 | Timer 2 Overflow Flag TF2 is not set if RCLK=1 or TCLK = 1. Must be cleared by software. Set by hardware on Timer 2 overflow. | | | | | |
| 6 | EXF2 | Timer 2 External Flag Set when a capture or a reload is caused by a negative transition on T2EX pin if EXEN2=1. Set to cause the CPU to vector to Timer 2 interrupt routine when Timer 2 interrupt is enabled. Must be cleared by software. | | | | | |
| 5 | RCLK | Receive Clock bit Clear to use timer 1 overflow as receive clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as receive clock for serial port in mode 1 or 3. | | | | | |
| 4 | TCLK | Transmit Clock bit Clear to use timer 1 overflow as transmit clock for serial port in mode 1 or 3. Set to use Timer 2 overflow as transmit clock for serial port in mode 1 or 3. | | | | | |
| 3 | EXEN2 | Timer 2 External Enable bit Clear to ignore events on T2EX pin for Timer 2 operation. Set to cause a capture or reload when a negative transition on T2EX pin is detected, if Timer 2 is not used to clock the serial port. | | | | | |
| 2 | TR2 | Timer 2 Run Control bit Clear to turn off Timer 2. Set to turn on Timer 2. | | | | | |
| 1 | C/T2# | Timer/Counter 2 Select bit Clear for timer operation (input from internal clock system: f_{OSC}). Set for counter operation (input from T2 input pin). | | | | | |
| 0 | CP/RL2# | Timer 2 Capture/Reload bit If RCLK=1 or TCLK=1, CP/RL2# is ignored and timer is forced to auto-reload on Timer 2 overflow. Clear to auto-reload on Timer 2 overflows or negative transitions on T2EX pin if EXEN2=1. Set to capture on negative transitions on T2EX pin if EXEN2=1. | | | | | |

Reset Value = 0000 0000b
bit addressable

Table 45. T2MOD Register
T2MOD (S:C9h)
Timer 2 Mode Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|------|------|
| - | - | - | - | - | - | T2OE | DCEN |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 2 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 1 | T2OE | Timer 2 Output Enable bit Clear to program P1.0/T2 as clock input or I/O port. Set to program P1.0/T2 as clock output. |
| 0 | DCEN | Down Counter Enable bit Clear to disable Timer 2 as up/down counter. Set to enable Timer 2 as up/down counter. |

Reset Value = XXXX XX00b
Not bit addressable

Table 46. TH2 Register
TH2 (S:CDh)
Timer 2 High Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|----------------------|
| 7 - 0 | | High Byte of Timer 2 |

Reset Value = 0000 0000b
Not bit addressable

Table 47. TL2 Register
TL2 (S:CCh)
Timer 2 Low Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---------------------|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | Low Byte of Timer 2 | | | | | |

Reset Value = 0000 0000b
Not bit addressable

Table 48. RCAP2H Register
RCAP2H (S:CBh)
Timer 2 Reload/Capture High Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--------------------------------------|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | High Byte of Timer 2 Reload/Capture. | | | | | |

Reset Value = 0000 0000b
Not bit addressable

Table 49. RCAP2L Register
RCAP2L (S:CAh) Timer 2 Reload/Capture Low Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|-------------------------------------|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | | Low Byte of Timer 2 Reload/Capture. | | | | | |

Reset Value = 0000 0000b
Not bit addressable

Watchdog Timer

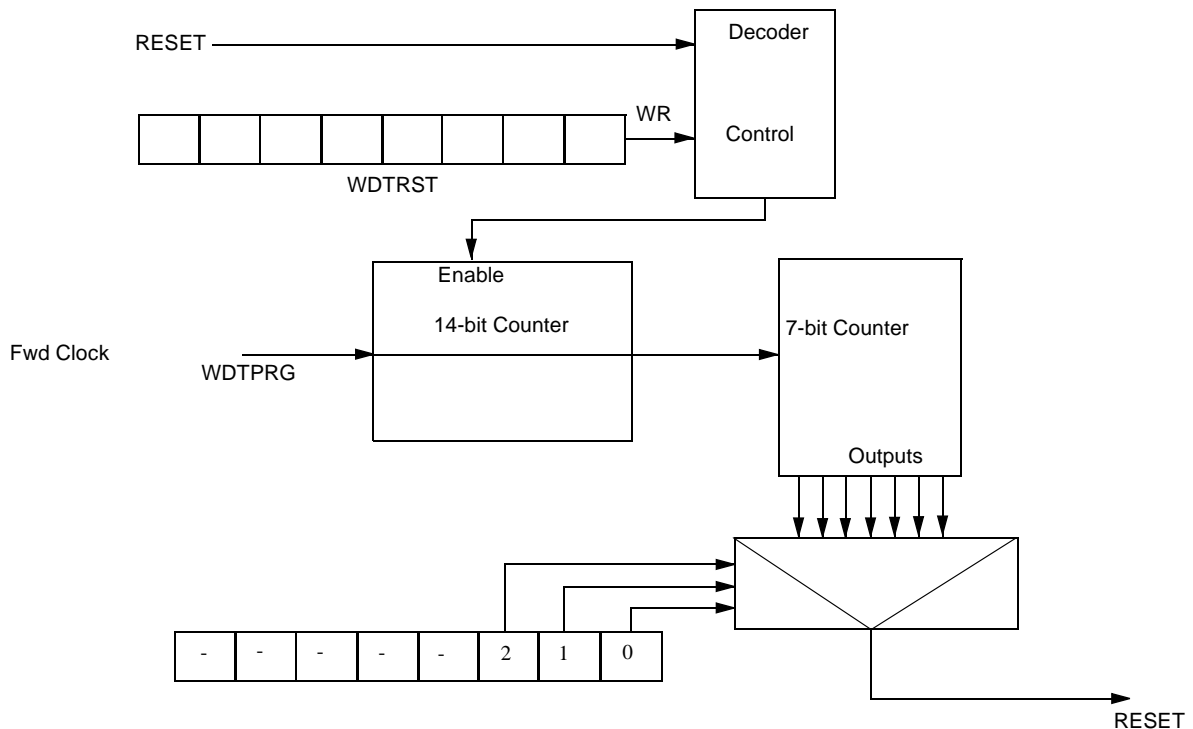
T89C51CC02 contains a powerful programmable hardware Watchdog Timer (WDT) that automatically resets the chip if it software fails to reset the WDT before the selected time interval has elapsed. It permits large Timeout ranging from 16ms to 2s @ $f_{OSC} = 12 \text{ MHz}$ in X1 mode.

This WDT consists of a 14-bit counter plus a 7-bit programmable counter, a Watchdog Timer reset register (WDTRST) and a Watchdog Timer programming (WDTPRG) register. When exiting reset, the WDT is -by default- disable.

To enable the WDT, the user has to write the sequence 1EH and E1H into WDTRST register with no instruction between the two writes. When the Watchdog Timer is enabled, it will increment every machine cycle while the oscillator is running and there is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is $96 \times T_{OSC}$, where $T_{OSC} = 1/f_{OSC}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset

Note: When the watchdog is enable it is impossible to change its period.

Figure 31. Watchdog Timer



Watchdog Programming

The three lower bits (S0, S1, S2) located into WDTPRG register permit to program the WDT duration.

Table 50. Machine Cycle Count

| S2 | S1 | S0 | Machine Cycle Count |
|----|----|----|---------------------|
| 0 | 0 | 0 | $2^{14} - 1$ |
| 0 | 0 | 1 | $2^{15} - 1$ |
| 0 | 1 | 0 | $2^{16} - 1$ |
| 0 | 1 | 1 | $2^{17} - 1$ |
| 1 | 0 | 0 | $2^{18} - 1$ |
| 1 | 0 | 1 | $2^{19} - 1$ |
| 1 | 1 | 0 | $2^{20} - 1$ |
| 1 | 1 | 1 | $2^{21} - 1$ |

To compute WD Timeout, the following formula is applied:

$$FTime - Out = \frac{F_{osc}}{6 \times 2^{WDX2 \wedge X2} (2^{14} \times 2^{Svalue})}$$

Note: Svalue represents the decimal value of (S2 S1 S0)

Find Hereafter computed Timeout values for $f_{oscXTAL} = 12 \text{ MHz}$ in X1 mode

Table 51. Timeout Computation

| S2 | S1 | S0 | $f_{osc}=12 \text{ MHz}$ | $f_{osc}=16\text{MHz}$ | $f_{osc}=20 \text{ MHz}$ |
|----|----|----|--------------------------|------------------------|--------------------------|
| 0 | 0 | 0 | 16.38 ms | 12.28 ms | 9.82 ms |
| 0 | 0 | 1 | 32.77 ms | 24.57 ms | 19.66 ms |
| 0 | 1 | 0 | 65.54 ms | 49.14 ms | 39.32 ms |
| 0 | 1 | 1 | 131.07 ms | 98.28 ms | 78.64 ms |
| 1 | 0 | 0 | 262.14 ms | 196.56 ms | 157.28 ms |
| 1 | 0 | 1 | 524.29 ms | 393.12 ms | 314.56 ms |
| 1 | 1 | 0 | 1.05 s | 786.24 ms | 629.12 ms |
| 1 | 1 | 1 | 2.10 s | 1.57 s | 1.25 s |

Watchdog Timer During Power-down Mode and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are 2 methods of exiting Power-down mode: by a hardware reset or via a level activated external interrupt which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, the watchdog is disabled. Exiting Power-down with an interrupt is significantly different. The interrupt shall be held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down.

To ensure that the WDT does not overflow within a few states of exiting powerdown, it is best to reset the WDT just before entering powerdown.

In the Idle mode, the oscillator continues to run. To prevent the WDT from resetting T89C51CC02 while in Idle mode, the user should always set up a timer that will periodically exit Idle, service the WDT, and re-enter Idle mode.

Register

Table 52. WDTPRG Register

WDTPRG (S:A7h) – Watchdog Timer Duration Programming register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|----|----|----|
| - | - | - | - | - | S2 | S1 | S0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 2 | S2 | Watchdog Timer Duration selection bit 2 Work in conjunction with bit 1 and bit 0. | | | | | |
| 1 | S1 | Watchdog Timer Duration selection bit 1 Work in conjunction with bit 2 and bit 0. | | | | | |
| 0 | S0 | Watchdog Timer Duration selection bit 0 Work in conjunction with bit 1 and bit 2. | | | | | |

Reset Value = XXXX X000b

Table 53. WDTRST Register
WDTRST (S:A6h Write Only) – Watchdog Timer Enable register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|------------------------|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Watchdog Control Value | | | | | |

Reset Value = 1111 1111b

Note: The WDRST register is used to reset/enable the WDT by writing 1EH then E1H in sequence without instruction between these two sequences.

CAN Controller

The CAN Controller provides all the features required to implement the serial communication protocol CAN as defined by BOSCH GmbH. The CAN specification as referred to by ISO/11898 (2.0A & 2.0B) for high speed and ISO/11519-2 for low speed. The CAN Controller is able to handle all types of frames (Data, Remote, Error and Overload) and achieves a bitrate of 1-Mbit/s at 8 MHz¹ Crystal frequency in X2 Mode.

Note: 1. At BRP = 1 sampling point will be fixed.

CAN Protocol

The CAN protocol is an international standard defined in the ISO 11898 for high speed and ISO 11519-2 for low speed.

Principles

CAN is based on a broadcast communication mechanism. This broadcast communication is achieved by using a message oriented transmission protocol. These messages are identified by using a message identifier. Such a message identifier has to be unique within the whole network and it defines not only the content but also the priority of the message.

The priority at which a message is transmitted compared to another less urgent message is specified by the identifier of each message. The priorities are laid down during system design in the form of corresponding binary values and cannot be changed dynamically. The identifier with the lowest binary number has the highest priority.

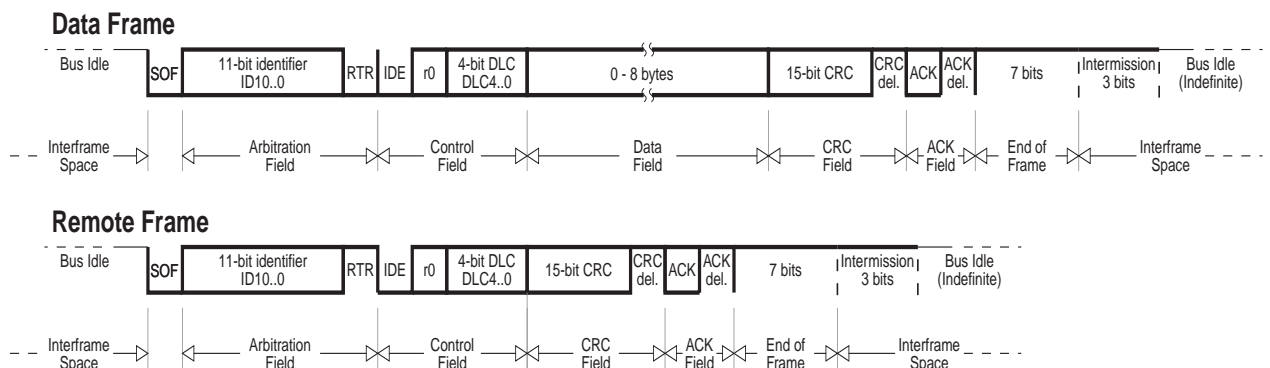
Bus access conflicts are resolved by bit-wise arbitration on the identifiers involved by each node observing the bus level bit for bit. This happens in accordance with the "wired and" mechanism, by which the dominant state overwrites the recessive state. The competition for bus allocation is lost by all nodes with recessive transmission and dominant observation. All the "losers" automatically become receivers of the message with the highest priority and do not re-attempt transmission until the bus is available again.

Message Formats

The CAN protocol supports two message frame formats, the only essential difference being in the length of the identifier. The CAN standard frame, also known as CAN 2.0 A, supports a length of 11 bits for the identifier, and the CAN extended frame, also known as CAN 2.0 B, supports a length of 29 bits for the identifier.

Can Standard Frame

Figure 32. CAN Standard Frames



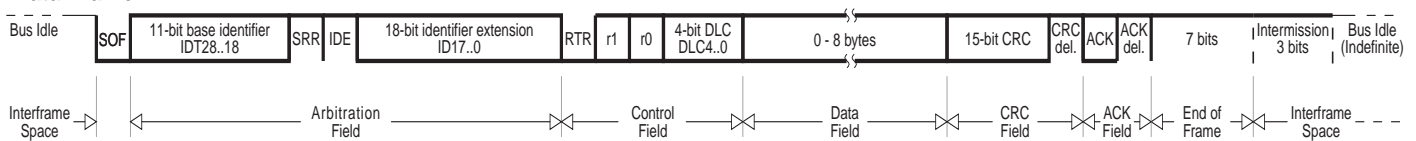
A message in the CAN standard frame format begins with the "Start Of Frame (SOF)", this is followed by the "Arbitration field" which consist of the identifier and the "Remote Transmission Request (RTR)" bit used to distinguish between the data frame and the data request frame called remote frame. The following "Control field" contains the "Identifier Extension (IDE)" bit and the "Data Length Code (DLC)" used to indicate the

number of following data bytes in the "Data field". In a remote frame, the DLC contains the number of requested data bytes. The "Data field" that follows can hold up to 8 data bytes. The frame integrity is guaranteed by the following "Cyclic Redundant Check (CRC)" sum. The "ACKnowledge (ACK) field" compromises the ACK slot and the ACK delimiter. The bit in the ACK slot is sent as a recessive bit and is overwritten as a dominant bit by the receivers which have at this time received the data correctly. Correct messages are acknowledged by the receivers regardless of the result of the acceptance test. The end of the message is indicated by "End Of Frame (EOF)". The "Intermission Frame Space (IFS)" is the minimum number of bits separating consecutive messages. If there is no following bus access by any node, the bus remains idle.

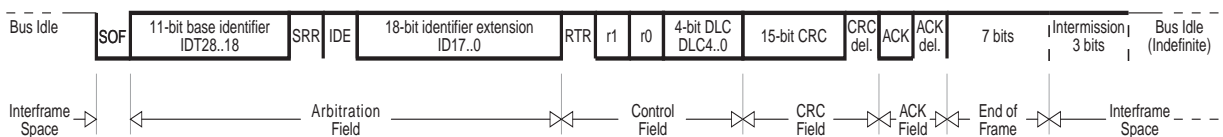
CAN Extended Frame

Figure 33. CAN Extended Frames

Data Frame



Remote Frame



A message in the CAN extended frame format is likely the same as a message in CAN standard frame format. The difference is the length of the identifier used. The identifier is made up of the existing 11-bit identifier (base identifier) and an 18-bit extension (identifier extension). The distinction between CAN standard frame format and CAN extended frame format is made by using the IDE bit which is transmitted as dominant in case of a frame in CAN standard frame format, and transmitted as recessive in the other case.

Format Co-existence

As the two formats have to co-exist on one bus, it is laid down which message has higher priority on the bus in the case of bus access collision with different formats and the same identifier / base identifier: The message in CAN standard frame format always has priority over the message in extended format.

There are three different types of CAN modules available:

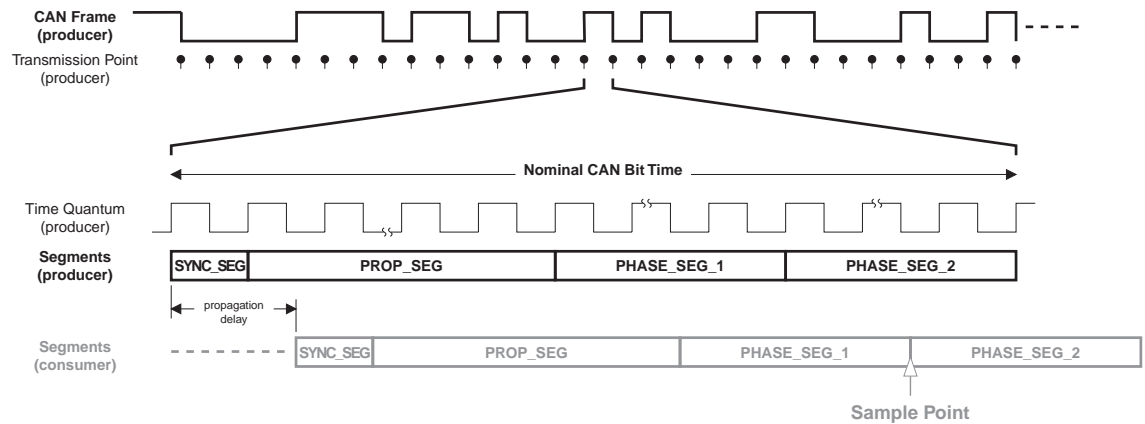
- 2.0A - Considers 29 bit ID as an error
- 2.0B Passive - Ignores 29 bit ID messages
- 2.0B Active - Handles both 11 and 29 bit ID Messages

Bit Timing

To ensure correct sampling up to the last bit, a CAN node needs to re-synchronize throughout the entire frame. This is done at the beginning of each message with the falling edge SOF and on each recessive to dominant edge.

Bit Construction

One CAN bit time is specified as four non-overlapping time segments. Each segment is constructed from an integer multiple of the Time Quantum. The Time Quantum or TQ is the smallest discrete timing resolution used by a CAN node.

Figure 34. CAN Bit Construction**Synchronization Segment**

The first segment is used to synchronize the various bus nodes.

On transmission, at the start of this segment, the current bit level is output. If there is a bit state change between the previous bit and the current bit, then the bus state change is expected to occur within this segment by the receiving nodes.

Propagation Time Segment

This segment is used to compensate for signal delays across the network.

This is necessary to compensate for signal propagation delays on the bus line and through the transceivers of the bus nodes.

Phase Segment 1

Phase Segment 1 is used to compensate for edge phase errors.

This segment may be lengthened during resynchronization.

Sample Point

The sample point is the point of time at which the bus level is read and interpreted as the value of the respective bit. Its location is at the end of Phase Segment 1 (between the two Phase Segments).

Phase Segment 2

This segment is also used to compensate for edge phase errors.

This segment may be shortened during resynchronization, but the length has to be at least as long as the information processing time and may not be more than the length of Phase Segment 1.

Information Processing Time

It is the time required for the logic to determine the bit level of a sampled bit.

The Information processing Time begins at the sample point, is measured in TQ and is fixed at 2 TQ for the Atmel CAN. Since Phase Segment 2 also begins at the sample point and is the last segment in the bit time, Phase Segment 2 minimum shall not be less than the Information processing Time.

Bit Lengthening

As a result of resynchronization, Phase Segment 1 may be lengthened or Phase Segment 2 may be shortened to compensate for oscillator tolerances. If, for example, the transmitter oscillator is slower than the receiver oscillator, the next falling edge used for resynchronization may be delayed. So Phase Segment 1 is lengthened in order to adjust the sample point and the end of the bit time.

Bit Shortening

If, on the other hand, the transmitter oscillator is faster than the receiver one, the next falling edge used for resynchronization may be too early. So Phase Segment 2 in bit N is shortened in order to adjust the sample point for bit N+1 and the end of the bit time

Synchronization Jump Width

The limit to the amount of lengthening or shortening of the Phase Segments is set by the Resynchronization Jump Width.

This segment may not be longer than Phase Segment 2.

Programming the Sample Point

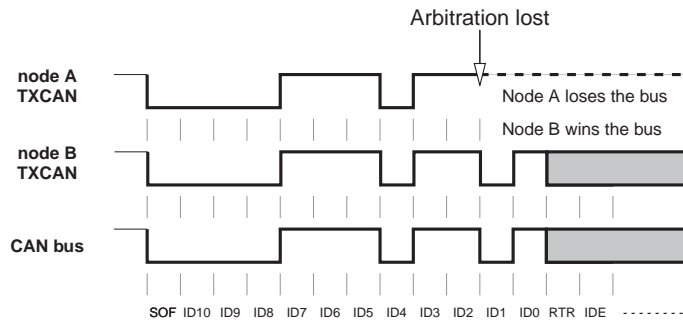
Programming of the sample point allows "tuning" of the characteristics to suit the bus.

Early sampling allows more Time Quanta in the Phase Segment 2 so the Synchronization Jump Width can be programmed to its maximum. This maximum capacity to shorten or lengthen the bit time decreases the sensitivity to node oscillator tolerances, so that lower cost oscillators such as ceramic resonators may be used.

Late sampling allows more Time Quanta in the Propagation Time Segment which allows a poorer bus topology and maximum bus length.

Arbitration

Figure 35. Bus Arbitration



The CAN protocol handles bus accesses according to the concept called "Carrier Sense Multiple Access with Arbitration on Message Priority".

During transmission, arbitration on the CAN bus can be lost to a competing device with a higher priority CAN Identifier. This arbitration concept avoids collisions of messages whose transmission was started by more than one node simultaneously and makes sure the most important message is sent first without time loss.

The bus access conflict is resolved during the arbitration field mostly over the identifier value. If a data frame and a remote frame with the same identifier are initiated at the same time, the data frame prevails over the remote frame (c.f. RTR bit).

Errors

The CAN protocol signals any errors immediately as they occur. Three error detection mechanisms are implemented at the message level and two at the bit level:

Error at Message Level

- **Cyclic Redundancy Check (CRC)**
The CRC safeguards the information in the frame by adding redundant check bits at the transmission end. At the receiver these bits are re-computed and tested against the received bits. If they do not agree there has been a CRC error.
- **Frame Check**
This mechanism verifies the structure of the transmitted frame by checking the bit

fields against the fixed format and the frame size. Errors detected by frame checks are designated "format errors".

- **ACK Errors**
As already mentioned frames received are acknowledged by all receivers through positive acknowledgement. If no acknowledgement is received by the transmitter of the message an ACK error is indicated.

Error at Bit Level

- **Monitoring**
The ability of the transmitter to detect errors is based on the monitoring of bus signals. Each node which transmits also observes the bus level and thus detects differences between the bit sent and the bit received. This permits reliable detection of global errors and errors local to the transmitter.
- **Bit Stuffing**
The coding of the individual bits is tested at bit level. The bit representation used by CAN is "Non Return to Zero (NRZ)" coding, which guarantees maximum efficiency in bit coding. The synchronization edges are generated by means of bit stuffing.

Error Signalling

If one or more errors are discovered by at least one node using the above mechanisms, the current transmission is aborted by sending an "error flag". This prevents other nodes accepting the message and thus ensures the consistency of data throughout the network. After transmission of an erroneous message that has been aborted, the sender automatically re-attempts transmission.

CAN Controller Description

The CAN controller accesses are made through SFR. Several operations are possible by SFR:

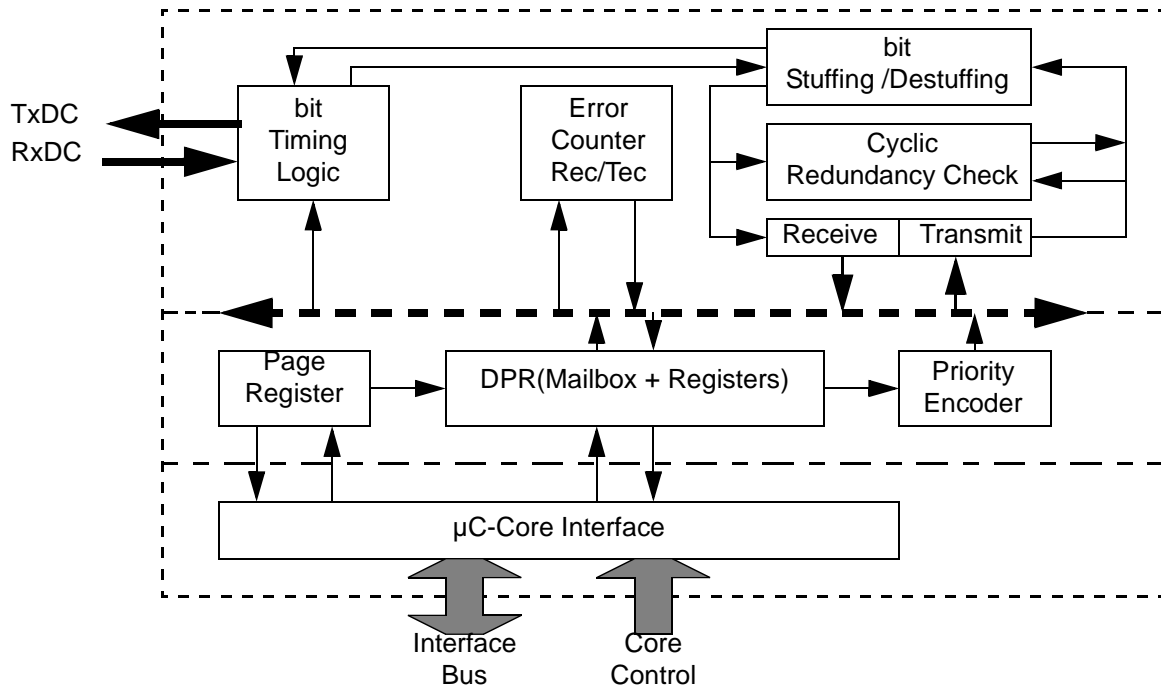
- arithmetic and logic operations, transfers and program control (SFR is accessible by direct addressing).
- 4 independent message objects are implemented, a pagination system manages their accesses.

Any message object can be programmed in a reception buffer block (even non-consecutive buffers). For the reception of defined messages one or several receiver message objects can be masked without participating in the buffer feature. An IT is generated when the buffer is full. The frames following the buffer-full interrupt will not be taken into account until at least one of the buffer message objects is re-enabled in reception. Higher priority of a message object for reception or transmission is given to the lower message object number.

The programmable 16-bit Timer (CANTIMER) is used to stamp each received and sent message in the CANSTMP register. This timer starts counting as soon as the CAN controller is enabled by the ENA bit in the CANGCON register.

The Time Trigger Communication (TTC) protocol is supported by the T89C51CC02.

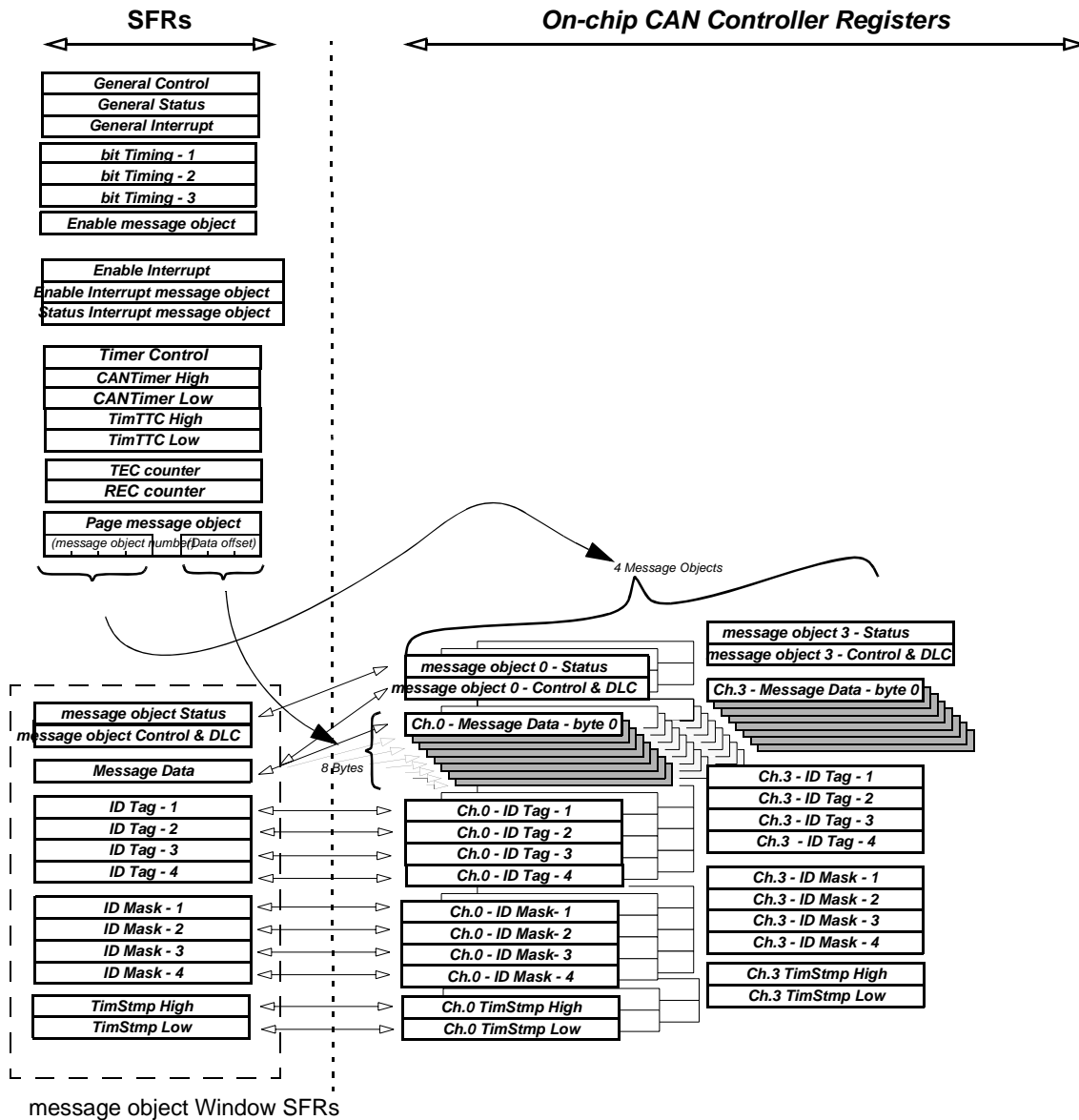
Figure 36. CAN Controller Block Diagram



CAN Controller Mailbox and Registers Organization

The pagination allows management of the 91 registers including 80(4 x 20) Bytes of mailbox via 32 SFRs. All actions on the message object window SFRs apply to the corresponding message object registers pointed by the message object number find in the Page message object register (CANPAGE) as illustrate in Figure 37.

Figure 37. CAN Controller Memory Organization



Working on Message Objects

The Page message object register (CANPAGE) is used to select one of the 4 message objects. Then, message object Control (CANCONCH) and message object Status (CANSTCH) are available for this selected message object number in the corresponding SFRs. A single register (CANMSG) is used for the message. The mailbox pointer is managed by the Page message object register with an auto-incrementation at the end of each access. The range of this counter is 8.

Note that the mailbox is a pure RAM, dedicated to one message object, without overlap. In most cases, it is not necessary to transfer the received message into the standard memory. The message to be transmitted can be built directly in the mailbox. Most calculations or tests can be executed in the mailbox area which provide quicker access.

CAN Controller Management

In order to enable the CAN Controller correctly the following registers have to be initialized:

- General Control (CANGCON),
- bit Timing (CANBT 1, 2 & 3),
- And for each page of 15 message objects:
 - Message object Control (CANCONCH),
 - Message object Status (CANSTCH).

During operation, the CAN Enable message object registers (CANEN) gives a fast overview of the message objects availability.

The CAN messages can be handled by interrupt or polling modes.

A message object can be configured as follows:

- Transmit message object
- Receive message object
- Receive buffer message object
- Disable

This configuration is made in the CONCH field of the CANCONCH register (See Table 54).

When a message object is configured, the corresponding ENCH bit of CANEN register is set.

Table 54. Configuration for CONCH1:2

| CONCH 1 | CONCH 2 | Type of Message Object |
|---------|---------|------------------------|
| 0 | 0 | Disable |
| 0 | 1 | Transmitter |
| 1 | 0 | Receiver |
| 1 | 1 | Receiver buffer |

When a Transmitter or Receiver action of a message object is completed, the corresponding ENCH bit of the CANEN register is cleared. In order to re-enable the message object, it is necessary to re-write the configuration in CANCONCH register.

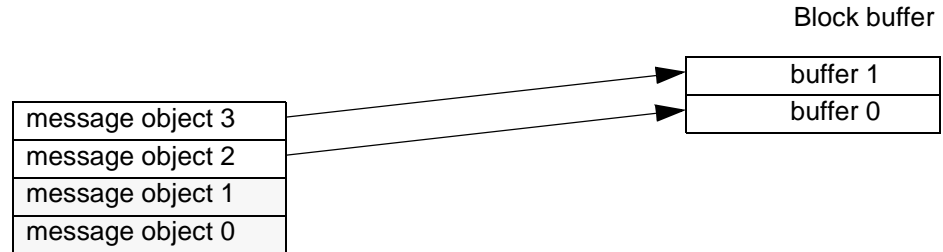
Non-consecutive message objects can be used for all three types of message objects (Transmitter, Receiver and Receiver buffer).

Buffer Mode

Any message object can be used to define one buffer, including non-consecutive message objects, and with no limitation in number of message objects used up to 4.

Each message object of the buffer must be initialized CONCH2 = 1 and CONCH1 = 1;

Figure 38. Buffer Mode



The same acceptance filter must be defined for each message objects of the buffer. When there is no mask on the identifier or the IDE, all messages are accepted.

A received frame will always be stored in the lowest free message object.

When the flag RxOk is set on one of the buffer message objects, this message object can then be read by the application. This flag must then be cleared by the software and the message object re-enabled in buffer reception in order to free the message object.

The OVRBUF flag in the CANGIT register is set when the buffer is full. This flag can generate an interrupt.

The frames following the buffer-full interrupt will not be stored and no status will be overwritten in the CANSTCH registers involved in the buffer until at least one of the buffer message objects is re-enabled in reception.

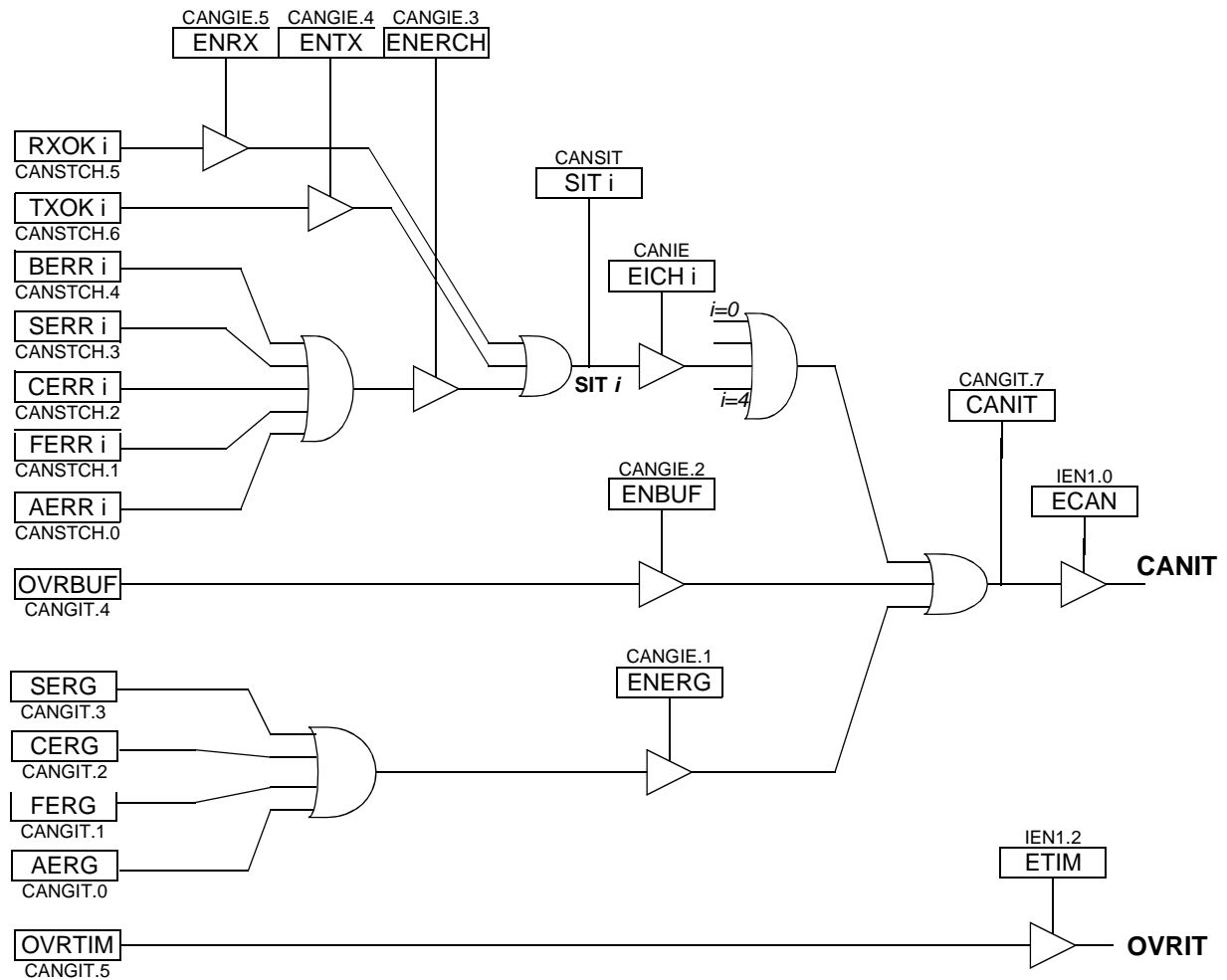
This flag must be cleared by the software in order to acknowledge the interrupt.

IT CAN Management

The different interrupts are:

- Transmission interrupt
- Reception interrupt
- Interrupt on error (bit error, stuff error, crc error, form error, acknowledge error)
- Interrupt when Buffer receive is full
- Interrupt on overrun of CAN Timer

Figure 39. CAN Controller Interrupt Structure



To enable a transmission interrupt:

- Enable General CAN IT in the interrupt system register
- Enable interrupt by message object, EICH_i
- Enable transmission interrupt, ENTX

To enable a reception interrupt:

- Enable General CAN IT in the interrupt system register
- Enable interrupt by message object, EICH_i
- Enable reception interrupt, ENRX

To enable an interrupt on message object error:

- Enable General CAN IT in the interrupt system register
- Enable interrupt by message object, EICH_i
- Enable interrupt on error, ENERCH

To enable an interrupt on general error:

- Enable General CAN IT in the interrupt system register
- Enable interrupt on error, ENERG

To enable an interrupt on Buffer-full condition:

- Enable General CAN IT in the interrupt system register
- Enable interrupt on Buffer full, ENBUF

To enable an interrupt when Timer overruns:

- Enable Overrun IT in the interrupt system register

When an interrupt occurs, the corresponding message object bit is set in the SIT register.

To acknowledge an interrupt, the corresponding CANSTCH bits (RXOK, TXOK,...) or CANGIT bits (OVRTIM, OVRBUF,...), must be cleared by the software application.

When the CAN node is in transmission and detects a Form Error in its frame, a bit Error will also be raised. Consequently, two consecutive interrupts can occur, both due to the same error.

When a message object error occurs and is set in CANSTCH register, no general error are set in CANGIE register.

Bit Timing and Baud Rate

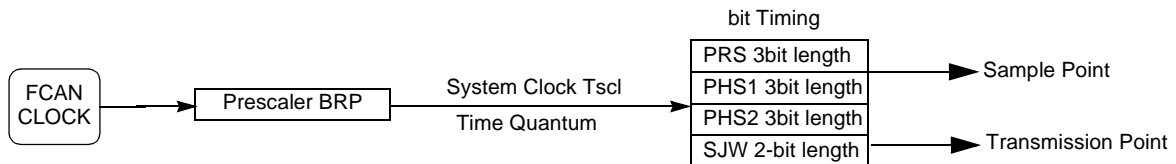
FSM's (Finite State Machine) of the CAN channel need to be synchronous to the time quantum. So, the input clock for bit timing is the clock used into CAN channel FSM's.

Field and segment abbreviations:

- BRP: Baud Rate Prescaler.
- TQ: Time Quantum (output of Baud Rate Prescaler).
- SYNS: SYNchronization Segment is 1 TQ long.
- PRS: PROpagation time Segment is programmable to be 1, 2, ..., 8 TQ long.
- PHS1: PHase Segment 1 is programmable to be 1, 2, ..., 8 TQ long.
- PHS2: PHase Segment 2 is programmable to be superior or equal to the Information Processing Time and inferior or equal to TPHS1
- INFORMATION PROCESSING TIME is 2 TQ.
- SJW: (Re) Synchronization Jump Width is programmable to be minimum of PHS1 and 4.

The total number of TQ in a bit time has to be programmed at least from 8 to 25.

Figure 40. Sample and Transmission Point



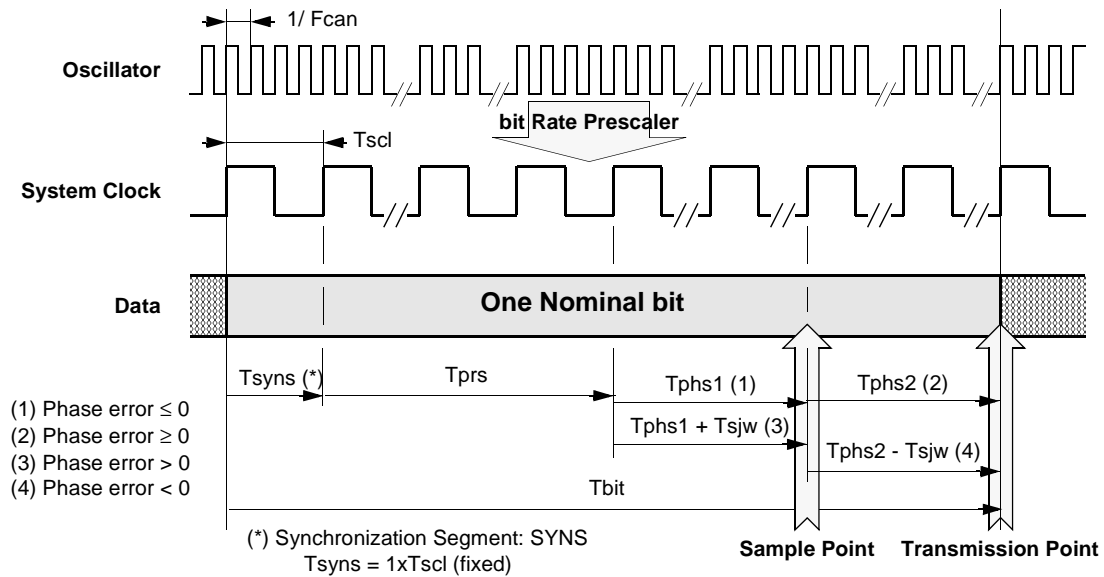
The baud rate selection is made by Tbit calculation:

$$T_{bit} = T_{syns} + T_{prs} + T_{phs1} + T_{phs2}$$

1. $T_{syns} = T_{scl} = (BRP[5..0] + 1) / F_{can} = 1TQ$
2. $T_{prs} = (1 \text{ to } 8) * T_{scl} = (PRS[2..0] + 1) * T_{scl}$
3. $T_{phs1} = (1 \text{ to } 8) * T_{scl} = (PHS1[2..0] + 1) * T_{scl}$
4. $T_{phs2} = (1 \text{ to } 8) * T_{scl} = (PHS2[2..0] + 1) * T_{scl}$
 $T_{phs2} = \text{Max of } (T_{phs1} \text{ and } 2TQ)$
5. $T_{sjw} = (1 \text{ to } 4) * T_{scl} = (SJW[1..0] + 1) * T_{scl}$

The total number of Tsc1 (Time Quanta) in a bit time must be comprised between 8 to 25.

Figure 41. General Structure of a bit Period



$$T_{bit} \text{ calculation: } T_{bit} = T_{syms} + T_{prs} + T_{phs1} + T_{phs2}$$

example of bit timing determination for CAN baudrate of 500 kbit/s:

$F_{osc} = 12 \text{ MHz}$ in X1 mode $\Rightarrow F_{CAN} = 6 \text{ MHz}$

Verify that the CAN baud rate you want is an integer division of FCAN clock.

$F_{CAN}/\text{CANbaudrate} = 6 \text{ MHz}/500 \text{ kHz} = 12$

The time quanta TQ must be comprised between 8 and 25: $TQ = 12$ and **BRP = 0**

Define the various timing parameters: $T_{bit} = T_{syms} + T_{prs} + T_{phs1} + T_{phs2} = 12TQ$

$T_{syms} = 1TQ$ and $T_{sjw} = 1TQ \Rightarrow \text{SJW} = 0$

If we chose a sample point at 66.6% $\Rightarrow T_{phs2} = 4TQ \Rightarrow \text{PHS2} = 3$

$T_{bit} = 12 = 4 + 1 + T_{phs1} + T_{prs}$, let us choose $T_{prs} = 3$ $T_{phs1} = 4$

PHS1 = 3 and PRS = 2

$\text{BRP} = 0$ so $\text{CANBT1} = 00h$

$\text{SJW} = 0$ and $\text{PRS} = 2$ so $\text{CANBT2} = 04h$

$\text{PHS2} = 3$ and $\text{PHS1} = 3$ so $\text{CANBT3} = 36h$

Fault Confinement

With respect to fault confinement, a unit may be in one of the three following status:

- Error active
- Error passive
- Bus off

An error active unit takes part in bus communication and can send an active error frame when the CAN macro detects an error.

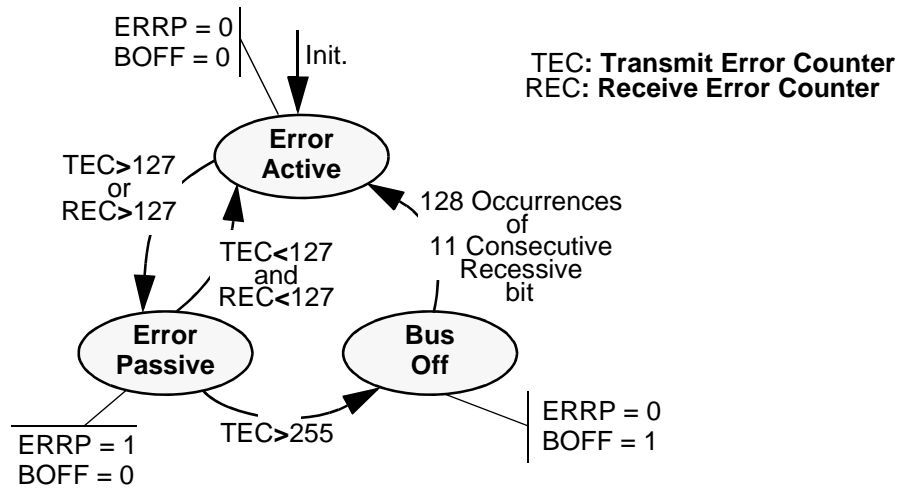
An error passive unit cannot send an active error frame. It takes part in bus communication, but when an error is detected, a passive error frame is sent. Also, after a transmission, an error passive unit will wait before initiating further transmission.

A bus off unit is not allowed to have any influence on the bus.

For fault confinement, two error counters (TEC and REC) are implemented.

See CAN Specification for details on Fault confinement.

Figure 42. Line Error Mode



Acceptance Filter

Upon a reception hit (i.e., a good comparison between the ID+RTR+RB+IDE received and an ID+RTR+RB+IDE specified while taking the comparison mask into account) the ID+RTR+RB+IDE received are written over the ID TAG Registers.

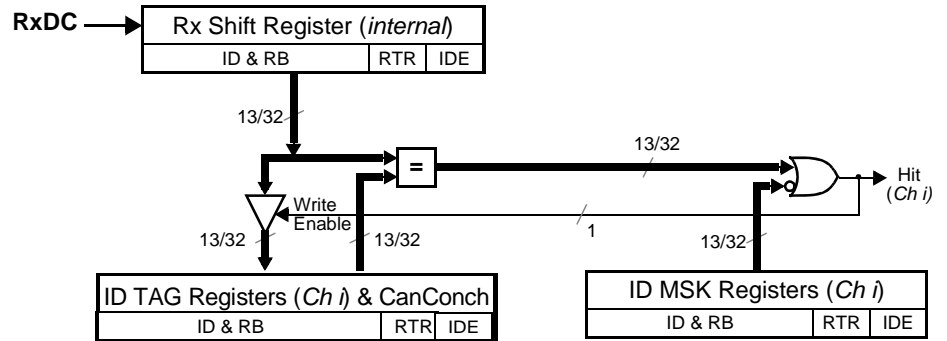
ID => IDT0-29

RTR => RTRTAG

RB => RB0-1TAG

IDE => IDE in CANCONCH register

Figure 43. Acceptance Filter Block Diagram



example:

To accept only ID = 318h in part A.

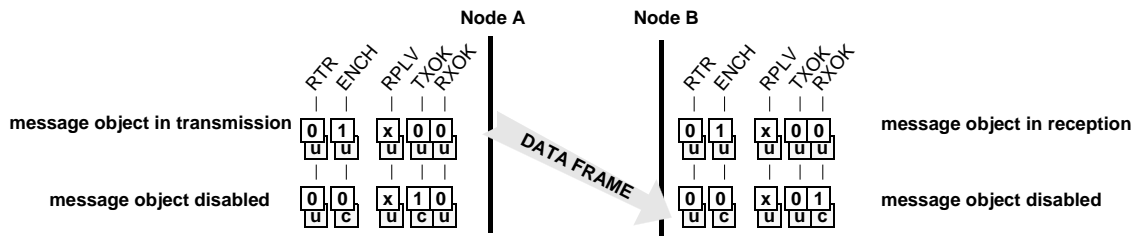
ID MSK = 111 1111 1111 b

ID TAG = 011 0001 1000 b

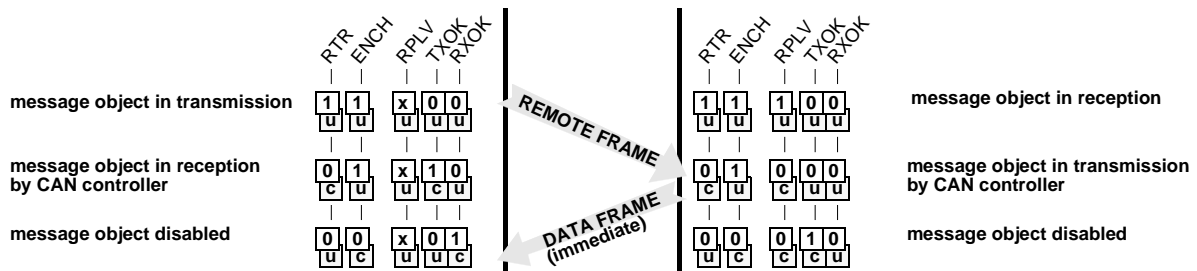
Data and Remote Frame

Description of the different steps for:

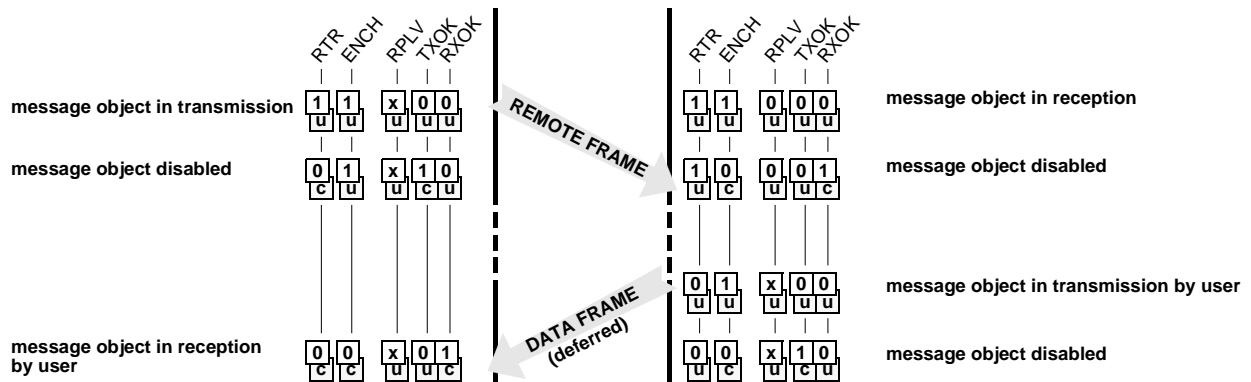
- Data frame



- Remote frame, with automatic reply



- Remote frame



| |
|---|
| i |
| u |

: modified by user

| |
|---|
| i |
| c |

: modified by CAN

Time Trigger Communication (TTC) and Message Stamping

The T89C51CC02 has a programmable 16-bit Timer (CANTIMH&CANTIML) for message stamp and TTC.

This CAN Timer starts after the CAN controller is enabled by the ENA bit in the CANGCON register.

Two modes in the timer are implemented:

- Time Trigger Communication:
 - Capture of this timer value in the CANTTCH & CANTTCL registers on Start Of Frame (SOF) or End Of Frame (EOF), depending on the SYNCTTC bit in the CANGCON register, when the network is configured in TTC by the TTC bit in the CANGCON register.

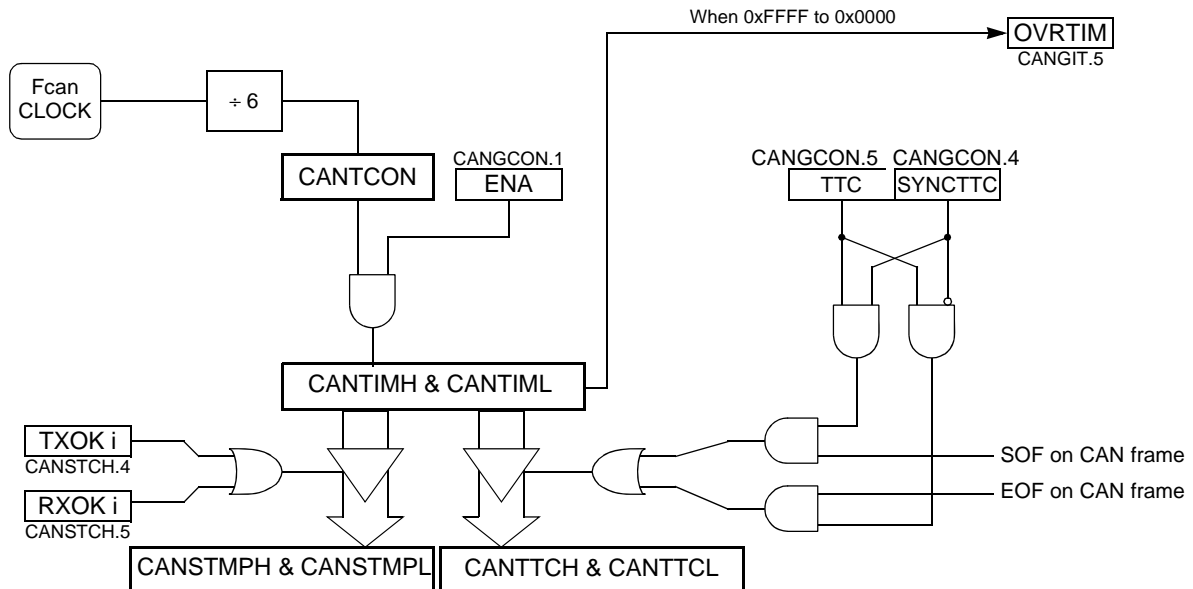
Note: In this mode, CAN **only sends the frame once, even if an error occurs**.

- Message Stamping
 - Capture of this timer value in the CANSTMPH & CANSTMPL registers of the message object which received or sent the frame.
 - All messages can be stamps.
 - The stamping of a received frame occurs when the RxOk flag is set.
 - The stamping of a sent frame occurs when the TxOk flag is set.

The CAN Timer works in a roll-over from FFFFh to 0000h which serves as a time base.

When the timer roll-over from FFFFh to 0000h, an interrupt is generated if the ETIM bit in the interrupt enable register IEN1 is set.

Figure 44. Block Diagram of CAN Timer



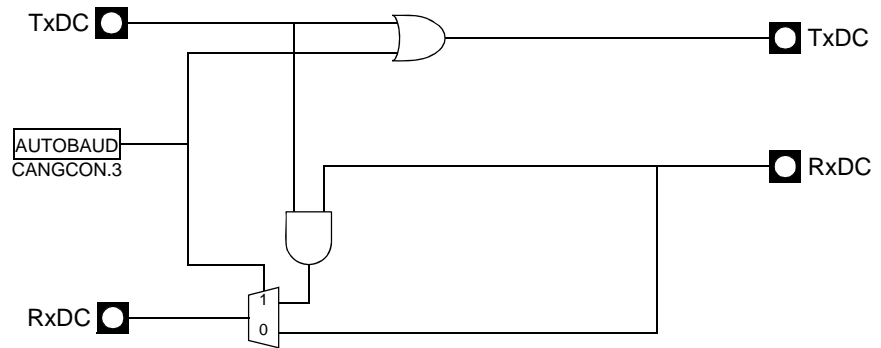
CAN Autobaud and Listening Mode

To activate the Autobaud feature, the AUTOBAUD bit in the CANGCON register must be set. In this mode, the CAN controller is only listening to the line without acknowledging the received messages. It cannot send any message. The error flags are updated. The bit timing can be adjusted until no error occurs (good configuration find).

In this mode, the error counters are frozen.

To go back to the standard mode, the AUTOBAUD bit must be cleared.

Figure 45. Autobaud Mode



Routine Examples

```
1. Init of CAN macro
// Reset the CAN macro
CANGCON = 01h;
// Disable CAN interrupts
ECAN    = 0;
ETIM    = 0;
// Init the Mailbox
for num_page = 0; num_page < 4; num_page++
{
    CANPAGE = num_channel << 4;
    CANCONCH = 00h;
    CANSTCH = 00h;
    CANIDT1 = 00h;
    CANIDT2 = 00h;
    CANIDT3 = 00h;
    CANIDT4 = 00h;
    CANIDM1 = 00h;
    CANIDM2 = 00h;
    CANIDM3 = 00h;
    CANIDM4 = 00h;
    for num_data = 0; num_data < 8; num_data++
    {
        CANMSG = 00h;
    }
}
// Configure the bit timing
CANBT1 = xxh;
CANBT2 = xxh;
CANBT3 = xxh;
```

```

// Enable the CAN macro
CANGCON = 02h

2. Configure message object 3 in reception to receive only standard (11bit
identifier) message 100h
// Select the message object 3
CANPAGE = 30h
// Enable the interrupt on this message object
CANIE = 08h
// Clear the status and control register
CANSTCH = 00h
CANCONCH= 00h
// Init the acceptance filter to accept only message 100h in standard mode
CANIDT1 = 20h
CANIDT2 = 00h
CANIDT3 = 00h
CANIDT4 = 00h
CANIDM1 = FFh
CANIDM2 = FFh
CANIDM3 = FFh
CANIDM4 = FFh
// Enable channel in reception
CANCONCH = 88h // enable reception

```

Note: to enable the CAN interrupt in reception:

```

EA = 1
ECAN = 1
CANGIE = 20h

3. Send a message on the message object 0
// Select the message object 0
CANPAGE = 00h
// Enable the interrupt on this message object
CANIE = 01h
// Clear the Status register
CANSTCH = 00h;
// load the identifier to send (ex: 555h)
CANIDT1 = AAh;
CANIDT2 = A0h;
// load data to send
CANMSG = 00h
CANMSG = 01h
CANMSG = 02h
CANMSG = 03h
CANMSG = 04h
CANMSG = 05h
CANMSG = 06h
CANMSG = 07h
// configure the control register
CANCONCH = 18h

4. Interrupt routine
// Save the current CANPAGE

```

```
// Find the first message object which generate an interrupt in CANSIT
// Select the corresponding message object

// Analyse the CANSTCH register to identify which kind of interrupt is
generated

// Manage the interrupt

// Clear the status register CANSTCH = 00h;

// if it is not a channel interrupt but a general interrupt
// Manage the general interrupt and clear CANGIT register

// restore the old CANPAGE
```

CAN SFRs

Table 55. SFR Mapping

| | 0/8 ⁽¹⁾ | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |
|-----|---------------------------|----------------------|-----------------------------------|-----------------------|----------------------|----------------------|-----------------------|-----------------------|-----|
| F8h | IPL1 xxxx x000 | CH 0000 0000 | CCAP0H 0000 0000 | CCAP1H 0000 0000 | | | | | FFh |
| F0h | B 0000 0000 | | ADCLK xxx0 0000 | ADCON x000 0000 | ADDL 0000 0000 | ADDH 0000 0000 | ADCF 0000 0000 | IPH1 xxxx x000 | F7h |
| E8h | IEN1 xxxx x000 | CL 0000 0000 | CCAP0L 0000 0000 | CCAP1L 0000 0000 | | | | | EFh |
| E0h | ACC 0000 0000 | | | | | | | | E7h |
| D8h | CCON 0000 0000 | CMOD 0xxx x000 | CCAPM0 x000 0000 | CCAPM1 x000 0000 | | | | | DFh |
| D0h | PSW 0000 0000 | FCON 0000 0000 | EECON xxxx xx00 | | | | | | D7h |
| C8h | T2CON 0000 0000 | T2MOD xxxx xx00 | RCAP2L 0000 0000 | RCAP2H 0000 0000 | TL2 0000 0000 | TH2 0000 0000 | | CANEN xxxx 0000 | CFh |
| C0h | P4 xxxx xx11 | CANGIE 1100 0000 | | CANIE 1111 0000 | CANIDM1 xxxx xxxx | CANIDM2 xxxx xxxx | CANIDM3 xxxx xxxx | CANIDM4 xxxx xxxx | C7h |
| B8h | IPL0 x000 0000 | SADEN 0000 0000 | | CANSIT xxxx 0000 | CANIDT1 xxxx xxxx | CANIDT2 xxxx xxxx | CANIDT3 xxxx xxxx | CANIDT4 xxxx xxxx | BFh |
| B0h | P3 1111 1111 | CANPAGE 1100 0000 | CANSTCH xxxx xxxx | CANCONCH xxxx xxxx | CANBT1 xxxx xxxx | CANBT2 xxxx xxxx | CANBT3 xxxx xxxx | IPH0 x000 0000 | B7h |
| A8h | IEN0 0000 0000 | SADDR 0000 0000 | CANGSTA 1010 0000 | CANGCON 0000 0000 | CANTIML 0000 0000 | CANTIMH 0000 0000 | CANSTMPL xxxx xxxx | CANSTMPH xxxx xxxx | AFh |
| A0h | P2 xxxx xx11 | CANTCON 0000 0000 | AUXR1 ⁽²⁾ xxxx 00x0 | CANMSG xxxx xxxx | CANTTCL 0000 0000 | CANTTCH 0000 0000 | WDTRST 1111 1111 | WDTPRG xxxx x000 | A7h |
| 98h | SCON 0000 0000 | SBUF 0000 0000 | | CANGIT 0x00 0000 | CANTEC 0000 0000 | CANREC 0000 0000 | | | 9Fh |
| 90h | P1 1111 1111 | | | | | | | | 97h |
| 88h | TCON 0000 0000 | TMOD 0000 0000 | TL0 0000 0000 | TL1 0000 0000 | TH0 0000 0000 | TH1 0000 0000 | | CKCON 0000 0000 | 8Fh |
| 80h | | SP 0000 0111 | DPL 0000 0000 | DPH 0000 0000 | | | | PCON 00x1 0000 | 87h |
| | 0/8 ⁽¹⁾ | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F | |

Registers

Table 56. CANGCON Register

CANGCON (S:ABh)
CAN General Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---------|----------|------|-----|------|
| ABRQ | OVRQ | TTC | SYNCTTC | AUTOBAUD | TEST | ENA | GRES |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | ABRQ | Abort Request Not an auto-resettable bit. A reset of the ENCH bit (message object control & DLC register) is done for each message object. The pending transmission communications are immediately aborted but the on-going communication will be terminated normally, setting the appropriate status flags, TxOk or RxOk. | | | | | |
| 6 | OVRQ | Overload Frame Request (Initiator). Auto-resettable bit. Set to send an overload frame after the next received message. Cleared by the hardware at the beginning of transmission of the overload frame. | | | | | |
| 5 | TTC | Network in Timer Trigger Communication set to select node in TTC. clear to disable TTC features. | | | | | |
| 4 | SYNCTTC | Synchronization of TTC When this bit is set the TTC timer is caught on the last bit of the End Of Frame. When this bit is clear the TTC timer is caught on the Start Of Frame. This bit is only used in the TTC mode. | | | | | |
| 3 | AUTOBAUD | AUTOBAUD set to active listening mode. Clear to disable listening mode | | | | | |
| 2 | TEST | Test mode. The test mode is intended for factory testing and not for customer use. | | | | | |
| 1 | ENA/STB | Enable/Standby CAN Controller When this bit is set, it enables the CAN controller and its input clock. When this bit is clear, the on-going communication is terminated normally and the CAN controller state of the machine is frozen (the ENCH bit of each message object does not change). In the standby mode, the transmitter constantly provides a recessive level; the receiver is not activated and the input clock is stopped in the CAN controller. During the disable mode, the registers and the mailbox remain accessible. Note that two clock periods are needed to start the CAN controller state of the machine. | | | | | |
| 0 | GRES | General Reset (Software Reset). Auto-resettable bit. This reset command is 'ORed' with the hardware reset in order to reset the controller. After a reset, the controller is disabled. | | | | | |

Reset Value = 0000 0000b

Table 57. CANGSTA Register

CANGSTA (S:AAh Read Only)
CAN General Status Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|------|---|------|------|------|------|------|
| - | OVFG | - | TBSY | RBSY | ENFG | BOFF | ERRP |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | - | Reserved The values read from this bit is indeterminate. Do not set this bit. |
| 6 | OVFG | Overload frame flag This status bit is set by the hardware as long as the produced overload frame is sent. This flag does not generate an interrupt |
| 5 | - | Reserved The values read from this bit is indeterminate. Do not set this bit. |
| 4 | TBSY | Transmitter busy This status bit is set by the hardware as long as the CAN transmitter generates a frame (remote, data, overload or error frame) or an ack field. This bit is also active during an InterFrame Spacing if a frame must be sent. This flag does not generate an interrupt. |
| 3 | RBSY | Receiver busy This status bit is set by the hardware as long as the CAN receiver acquires or monitors a frame. This flag does not generate an interrupt. |
| 2 | ENFG | Enable on-chip CAN controller flag Because an enable/disable command is not effective immediately, this status bit gives the true state of a chosen mode. This flag does not generate an interrupt. |
| 1 | BOFF | Bus off mode See Figure 42 |
| 0 | ERRP | Error passive mode See Figure 42 |

Reset Value = x0x0 0000b

Table 58. CANGIT Register

CANGIT (S:9Bh)
CAN General Interrupt

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|--------|------|------|------|------|
| CANIT | - | OVRTIM | OVRBUF | SERG | CERG | FERG | AERG |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | CANIT | General interrupt flag⁽¹⁾ This status bit is the image of all the CAN controller interrupts sent to the interrupt controller. It can be used in the case of the polling method. | | | | | |
| 6 | - | Reserved The values read from this bit is indeterminate. Do not set this bit. | | | | | |
| 5 | OVRTIM | Overrun CAN Timer This status bit is set when the CAN timer switches 0xFFFF to 0x0000. If the bit ETIM in the IE1 register is set, an interrupt is generated. Clear this bit in order to reset the interrupt. | | | | | |
| 4 | OVRBUF | Overrun BUFFER 0 - no interrupt. 1 - IT turned on This bit is set when the buffer is full. bit resetable by user. See Figure 39. | | | | | |
| 3 | SERG | Stuff Error General Detection of more than five consecutive bits with the same polarity. This flag can generate an interrupt. resetable by user. | | | | | |
| 2 | CERG | CRC Error General The receiver performs a CRC check on each destuffed received message from the start of frame up to the data field. If this checking does not match with the destuffed CRC field, a CRC error is set. This flag can generate an interrupt. resetable by user. | | | | | |
| 1 | FERG | Form Error General The form error results from one or more violations of the fixed form in the following bit fields: CRC delimiter acknowledgment delimiter end_of_frame This flag can generate an interrupt. resetable by user. | | | | | |
| 0 | AERG | Acknowledgment Error General No detection of the dominant bit in the acknowledge slot. This flag can generate an interrupt. resetable by user. | | | | | |

Note: 1. This field is Read Only.

Reset Value = 0x00 0000b

Table 59. CANTEC Register
CANTEC (S:9Ch Read Only) – CAN Transmit Error Counter

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|------|--------------|------|---|------|------|------|
| TEC7 | TEC6 | TEC5 | TEC4 | TEC3 | TEC2 | TEC1 | TEC0 |
| Bit Number | | Bit Mnemonic | | Description | | | |
| 7 - 0 | | TEC7:0 | | Transmit Error Counter See Figure 42 | | | |

Reset Value = 00h

Table 60. CANREC Register
CANREC (S:9Dh Read Only) – CAN Reception Error Counter

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|------|--------------|------|--|------|------|------|
| REC7 | REC6 | REC5 | REC4 | REC3 | REC2 | REC1 | REC0 |
| Bit Number | | Bit Mnemonic | | Description | | | |
| 7 - 0 | | REC7:0 | | Reception Error Counter See Figure 42 | | | |

Reset Value = 00h

Table 61. CANGIE Register

CANGIE (S:C1h) – CAN

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|------|------|--------|-------|-------|---|
| - | - | ENRX | ENTX | ENERCH | ENBUF | ENERG | - |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 6 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 5 | ENRX | Enable Receive Interrupt 0 - Disable 1 - Enable |
| 4 | ENTX | Enable Transmit Interrupt 0 - Disable 1 - Enable |
| 3 | ENERCH | Enable Message Object Error Interrupt 0 - Disable 1 - Enable |
| 2 | ENBUF | Enable BUF Interrupt 0 - Disable 1 - Enable |
| 1 | ENERG | Enable General Error Interrupt 0 - Disable 1 - Enable |
| 0 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. See Figure 39. |

Reset Value = xx00 000xb

Table 62. CANEN Register
CANEN (S:CFh Read Only)
CAN Enable Message Object Registers

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|-------|-------|-------|-------|
| - | - | - | - | ENCH3 | ENCH2 | ENCH1 | ENCH0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 4 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 3 - 0 | ENCH3:0 | Enable Message Object 0 - message object is disabled => the message object is free for a new emission or reception. 1 - message object is enabled. This bit is resettable by re-writing the CANCONCH of the corresponding message object. |

Reset Value = xxxx 0000b

Table 63. CANSIT Register
CANSIT (S:BBh Read Only) – CAN Status Interrupt Message Object Registers

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|------|------|------|------|
| - | - | - | - | SIT3 | SIT2 | SIT1 | SIT0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 4 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 3 - 0 | SIT3:0 | Status of Interrupt by Message Object 0 - no interrupt. 1 - IT turned on. Reset when interrupt condition is cleared by user. SIT3:0 = 0b 0000 1001 -> IT's on message objects 3 & 0. See Figure 39. |

Reset Value = xxxx0000b

Table 64. CANIE Register
CANIE (S:C3h) – CAN Enable Interrupt message object Registers

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|--------|--------|--------|--------|
| - | - | - | - | IECH 3 | IECH 2 | IECH 1 | IECH 0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 4 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 3 - 0 | IECH3:0 | Enable Interrupt by Message Object 0 - disable IT. 1 - enable IT. IECH3:0 = 0b 0000 1100 -> Enable IT's of message objects 3 & 2. |

Reset Value = xxxx 0000b

Table 65. CANBT1 Register
CANBT1 (S:B4h) – CAN bit Timing Registers 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-------|-------|-------|-------|---|
| - | BRP 5 | BRP 4 | BRP 3 | BRP 2 | BRP 1 | BRP 0 | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 - 1 | BRP5:0 | Baud Rate Prescaler The period of the CAN controller system clock T _{scl} is programmable and determines the individual bit timing. ⁽¹⁾ $T_{scl} = \frac{BRP[5..0] + 1}{F_{CAN}}$ | | | | | |
| 0 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |

Note: 1. The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0.
See Figure 41.

No default value after reset.

Table 66. CANBT2 Register
CANBT2 (S:B5h) – CAN bit Timing Registers 2

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|-------|-------|---|-------|-------|-------|---|
| - | SJW 1 | SJW 0 | - | PRS 2 | PRS 1 | PRS 0 | - |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 6 - 5 | SJW1:0 | Re-synchronization Jump Width To compensate for phase shifts between clock oscillators of different bus controllers, the controller must re-synchronize on any relevant signal edge of the current transmission. The synchronization jump width defines the maximum number of clock cycles. A bit period may be shortened or lengthened by a re-synchronization. $T_{sjw} = T_{scl} \times (SJW [1..0] + 1)$ |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 - 1 | PRS2:0 | Programming Time Segment This part of the bit time is used to compensate for the physical delay times within the network. It is twice the sum of the signal propagation time on the bus line, the input comparator delay and the output driver delay. $T_{prs} = T_{scl} \times (PRS[2..0] + 1)$ |
| 0 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |

Note: 1. The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0.
See Figure 41.

No default value after reset.

Table 67. CANBT3 Register
CANBT3 (S:B6h)
CAN bit Timing Registers 3

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--------|--|--------|--------|--------|-----|
| - | PHS2 2 | PHS2 1 | PHS2 0 | PHS1 2 | PHS1 1 | PHS1 0 | SMP |
| Bit Number | Bit Mnemonic | | Description | | | | |
| 7 | - | | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | |
| 6 - 4 | PHS2 2:0 | | Phase Segment 2 This phase is used to compensate for phase edge errors. This segment can be shortened by the re-synchronization jump width. $T_{phs2} = T_{scl} \times (PHS2[2..0] + 1)$ Phase segment 2 is the maximum of Phase segment1 and the Information Processing Time (= 2TQ). | | | | |
| 3 - 1 | PHS1 2:0 | | Phase Segment 1 This phase is used to compensate for phase edge errors. This segment can be lengthened by the re-synchronization jump width. $T_{phs1} = T_{scl} \times (PHS1[2..0] + 1)$ | | | | |
| 0 | SMP | | Sample Type 0 - once, at the sample point. 1 - three times, the threefold sampling of the bus is the sample point and twice over a distance of a 1/2 period of the Tsc1. The result corresponds to the majority decision of the three values. | | | | |

Note: 1. The CAN controller bit timing registers must be accessed only if the CAN controller is disabled with the ENA bit of the CANGCON register set to 0.
See Figure 41.

No default value after reset.

Table 68. CANPAGE Register
CANPAGE (S:B1h) – CAN Message Object Page Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|--------|--------|------|-------|-------|-------|
| - | - | CHNB 1 | CHNB 0 | AINC | INDX2 | INDX1 | INDX0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 6 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 5 - 4 | CHNB3:0 | Selection of Message Object Number The available numbers are: 0 to 3(See Figure 37). |
| 3 | AINC | Auto Increment of the Index (Active Low) 0 - auto-increment of the index (default value). 1 - non-auto-increment of the index. |
| 2 - 0 | INDX2:0 | Index Byte location of the data field for the defined message object (See Figure 37). |

Reset Value = xx00 0000b

Table 69. CANCONCH Register
CANCONCH (S:B3h) – CAN Message Object Control and DLC Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|---------|------|-----|-------|-------|-------|-------|
| CONCH 1 | CONCH 0 | RPLV | IDE | DLC 3 | DLC 2 | DLC 1 | DLC 0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 6 | CONCH1:0 | Configuration of Message Object CONCH1 CONCH0 0 0: disable 0 1: Launch transmission 1 0: Enable Reception 1 1: Enable Reception Buffer NOTE: The user must re-write the configuration to enable the corresponding bit in the CANEN1:2 registers. |
| 5 | RPLV | Reply valid Used in the automatic reply mode after receiving a remote frame 0 - reply not ready. 1 - reply ready & valid. |
| 4 | IDE | Identifier Extension 0 - CAN standard rev 2.0 A (ident = 11 bits). 1 - CAN standard rev 2.0 B (ident = 29 bits). |
| 3 - 0 | DLC3:0 | Data Length Code Number of Bytes in the data field of the message. The range of DLC is from 0 up to 8. This value is updated when a frame is received (data or remote frame). If the expected DLC differs from the incoming DLC, a warning appears in the CANSTCH register. |

No default value after reset

Table 70. CANSTCH Register
CANSTCH (S:B2h) – CAN Message Object Status Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|------|------|------|------|------|
| DLCW | TXOK | RXOK | BERR | SERR | CERR | FERR | AERR |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | DLCW | Data Length Code Warning The incoming message does not have the DLC expected. Whatever the frame type, the DLC field of the CANCONCH register is updated by the received DLC. | | | | | |
| 6 | TXOK | Transmit OK The communication enabled by transmission is completed. When the controller is ready to send a frame, if two or more message objects are enabled as producers, the lower index message object (0 to 13) is supplied first. Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 5 | RXOK | Receive OK The communication enabled by reception is completed. In the case of two or more message object reception hits, the lower index message object (0 to 13) is updated first. Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 4 | BERR | bit Error (only in transmission) The bit value monitored is different from the bit value sent. Exceptions: the monitored recessive bit sent as a dominant bit during the arbitration field and the acknowledge slot detecting a dominant bit during the sending of an error frame. Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 3 | SERR | Stuff Error Detection of more than five consecutive bits with the same polarity. Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 2 | CERR | CRC Error The receiver performs a CRC check on each destuffed received message from the start of frame up to the data field. If this checking does not match with the destuffed CRC field, a CRC error is set. Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 1 | FERR | Form Error The form error results from one or more violations of the fixed form in the following bit fields: CRC delimiter acknowledgment delimiter end_of_frame Must be cleared by software. This flag can generate an interrupt. | | | | | |
| 0 | AERR | Acknowledgment Error No detection of the dominant bit in the acknowledge slot. Must be cleared by software. This flag can generate an interrupt. | | | | | |

Note: See Figure 39.

No default value after reset.

Table 71. CANIDT1 Register for V2.0 part A
CANIDT1 for V2.0 part A (S:BCh) – CAN Identifier Tag Registers 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-------|-------|-------|-------|-------|
| IDT 10 | IDT 9 | IDT 8 | IDT 7 | IDT 6 | IDT 5 | IDT 4 | IDT 3 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | IDT10:3 | Identifier Tag Value See Figure 43. | | | | | |

No default value after reset.

Table 72. CANIDT2 Register for V2.0 part A
CANIDT2 for V2.0 part A (S:BDh) – CAN Identifier Tag Registers 2

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|---|---|---|
| IDT 2 | IDT 1 | IDT 0 | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 5 | IDT2:0 | Identifier Tag Value See Figure 43. | | | | | |
| 4-0 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. | | | | | |

No default value after reset.

Table 73. CANIDT3 Register for V2.0 part A
CANIDT3 for V2.0 part A (S:BEh) –CAN Identifier Tag Registers 3

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. | | | | | |

No default value after reset.

Table 74. CANIDT1 for V2.0 part A

CANIDT4 for V2.0 part A (S:BFh)
CAN Identifier Tag Registers 4

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|--------|---|--------|
| - | - | - | - | - | RTRTAG | - | RB0TAG |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 3 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 2 | RTRTAG | Remote transmission request tag value. |
| 1 | - | Reserved The values read from this bit are indeterminate. Do not set these bit. |
| 0 | RB0TAG | Reserved bit 0 tag value. |

No default value after reset.

Table 75. CANIDT2Register for V2.0 part A

CANIDT1 for V2.0 Part B (S:BCh)
CAN Identifier Tag Registers 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|--------|--------|--------|--------|--------|--------|
| IDT 28 | IDT 27 | IDT 26 | IDT 25 | IDT 24 | IDT 23 | IDT 22 | IDT 21 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 0 | IDT28:21 | Identifier Tag Value See Figure 43. |

No default value after reset.

Table 76. CANIDT2 Register for V2.0 Part B

CANIDT2 for V2.0 Part B (S:BDh)
CAN Identifier Tag Registers 2

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|--------|--------|--------|--------|--------|--------|
| IDT 20 | IDT 19 | IDT 18 | IDT 17 | IDT 16 | IDT 15 | IDT 14 | IDT 13 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 0 | IDT20:13 | Identifier Tag Value See Figure 43. |

No default value after reset.

Table 77. CANIDT3 Register for V2.0 Part B
 CANIDT3 for V2.0 Part B (S:BEh)
 CAN Identifier Tag Registers 3

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-------|-------|-------|-------|-------|
| IDT 12 | IDT 11 | IDT 10 | IDT 9 | IDT 8 | IDT 7 | IDT 6 | IDT 5 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | IDT12:5 | Identifier Tag Value See Figure 43. | | | | | |

No default value after reset.

Table 78. CANIDT4 Register for V2.0 Part B
 CANIDT4 for V2.0 Part B (S:BFh)
 CAN Identifier Tag Registers 4

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-------|-------|--------|--------|--------|
| IDT 4 | IDT 3 | IDT 2 | IDT 1 | IDT 0 | RTRTAG | RB1TAG | RB0TAG |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 3 | IDT4:0 | Identifier Tag Value See Figure 43. | | | | | |
| 2 | RTRTAG | Remote Transmission Request Tag Value | | | | | |
| 1 | RB1TAG | Reserved bit 1 tag value. | | | | | |
| 0 | RB0TAG | Reserved bit 0 tag value. | | | | | |

No default value after reset.

Table 79. CANIDM1 Register for V2.0 part A
CANIDM1 for V2.0 part A (S:C4h)
CAN Identifier Mask Registers 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---------|---------|---------|---------|---------|
| IDMSK 10 | IDMSK 9 | IDMSK 8 | IDMSK 7 | IDMSK 6 | IDMSK 5 | IDMSK 4 | IDMSK 3 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | IDTMSK10:3 | Identifier Mask Value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. | | | | | |

No default value after reset.

Table 80. CANIDM2 Register for V2.0 part A
CANIDM2 for V2.0 part A (S:C5h)
CAN Identifier Mask Registers 2

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---|---|---|---|---|
| IDMSK 2 | IDMSK 1 | IDMSK 0 | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 5 | IDTMSK2:0 | Identifier Mask Value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. | | | | | |
| 4 - 0 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. | | | | | |

No default value after reset.

Table 81. CANIDM3 Register for V2.0 part A
CANIDM3 for V2.0 part A (S:C6h)
CAN Identifier Mask Registers 3

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | - | Reserved The values read from these bits are indeterminate. | | | | | |

No default value after reset.

Table 82. CANIDM4 Register for V2.0 part A
 CANIDM4 for V2.0 part A (S:C7h)
 CAN Identifier Mask Registers 4

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|--------|---|--------|
| - | - | - | - | - | RTRMSK | - | IDEMSK |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 3 | - | Reserved The values read from these bits are indeterminate. Do not set these bits. |
| 2 | RTRMSK | Remote transmission request Mask Value 0 - comparison true forced. 1 - bit comparison enabled. |
| 1 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 0 | IDEMSK | IDentifier Extension Mask Value 0 - comparison true forced. 1 - bit comparison enabled. |

Note: The ID Mask is only used for reception.

No default value after reset.

Table 83. CANIDM1 Register for V2.0 Part B
 CANIDM1 for V2.0 Part B (S:C4h)
 CAN Identifier Mask Registers 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| IDMSK 28 | IDMSK 27 | IDMSK 26 | IDMSK 25 | IDMSK 24 | IDMSK 23 | IDMSK 22 | IDMSK 21 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 0 | IDMSK28:21 | IDentifier Mask Value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. |

Note: The ID Mask is only used for reception.

No default value after reset.

Table 84. CANIDM2 Register for V2.0 Part B
CANIDM2 for V2.0 Part B (S:C5h)
CAN Identifier Mask Registers 2

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|----------|----------|----------|----------|----------|
| IDMSK 20 | IDMSK 19 | IDMSK 18 | IDMSK 17 | IDMSK 16 | IDMSK 15 | IDMSK 14 | IDMSK 13 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | IDMSK20:13 | Identifier Mask Value⁽¹⁾ 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. | | | | | |

Note: 1. The ID Mask is only used for reception.

No default value after reset.

Table 85. CANIDM3 Register for V2.0 Part B
CANIDM3 for V2.0 Part B (S:C6h)
CAN Identifier Mask Registers 3

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---------|---------|---------|---------|---------|
| IDMSK 12 | IDMSK 11 | IDMSK 10 | IDMSK 9 | IDMSK 8 | IDMSK 7 | IDMSK 6 | IDMSK 5 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | IDMSK12:5 | Identifier Mask Value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. | | | | | |

Note: The ID Mask is only used for reception.

No default value after reset.

Table 86. CANIDM4 Register for V2.0 Part B
CANIDM4 for V2.0 Part B (S:C7h)
CAN Identifier Mask Registers 4

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---------|---------|--------|---|--------|
| IDMSK 4 | IDMSK 3 | IDMSK 2 | IDMSK 1 | IDMSK 0 | RTRMSK | - | IDEMSK |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 3 | IDMSK4:0 | Identifier Mask Value 0 - comparison true forced. 1 - bit comparison enabled. See Figure 43. | | | | | |
| 2 | RTRMSK | Remote transmission request Mask Value 0 - comparison true forced. 1 - bit comparison enabled. | | | | | |
| 1 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 0 | IDEMSK | Identifier Extension Mask Value 0 - comparison true forced. 1 - bit comparison enabled. | | | | | |

Note: The ID Mask is only used for reception.

No default value after reset.

Table 87. CANMSG Register
CANMSG (S:A3h)
CAN Message Data Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-------|-------|-------|-------|-------|
| MSG 7 | MSG 6 | MSG 5 | MSG 4 | MSG 3 | MSG 2 | MSG 1 | MSG 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | MSG7:0 | Message Data This register contains the mailbox data byte pointed at the page message object register. After writing in the page message object register, this byte is equal to the specified message location (in the mailbox) of the pre-defined identifier + index. If auto-incrementation is used, at the end of the data register writing or reading cycle, the mailbox pointer is auto-incremented. The range of the counting is 8 with no end loop (0, 1,..., 7, 0,...) | | | | | |

No default value after reset.

Table 88. CANTCON Register
CANTCON (S:A1h)
CAN Timer ClockControl

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|----------|----------|----------|----------|----------|----------|----------|
| TPRESC 7 | TPRESC 6 | TPRESC 5 | TPRESC 4 | TPRESC 3 | TPRESC 2 | TPRESC 1 | TPRESC 0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 0 | TPRESC7:0 | Timer Prescaler of CAN Timer This register is a prescaler for the main timer upper counter range = 0 to 255. See Figure 44. |

Reset Value = 00h

Table 89. CANTIMH Register
CANTIMH (S:ADh)
CAN Timer High

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|------------|------------|------------|------------|------------|-----------|-----------|
| CANGTIM 15 | CANGTIM 14 | CANGTIM 13 | CANGTIM 12 | CANGTIM 11 | CANGTIM 10 | CANGTIM 9 | CANGTIM 8 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|---|
| 7 - 0 | CANGTIM15:8 | High byte of Message Timer See Figure 44. |

Reset Value = 0000 0000b

Table 90. CANTIML Register
CANTIML (S:ACh)
CAN Timer Low

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| CANGTIM 7 | CANGTIM 6 | CANGTIM 5 | CANGTIM 4 | CANGTIM 3 | CANGTIM 2 | CANGTIM 1 | CANGTIM 0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 - 0 | CANGTIM7:0 | Low byte of Message Timer See Figure 44. |

Reset Value = 0000 0000b

Table 91. CANSTMPH Register
CANSTMPH (S:AFh Read Only)
CAN Stamp Timer High

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|------------|------------|------------|-----------|-----------|
| TIMSTMP 15 | TIMSTMP 14 | TIMSTMP 13 | TIMSTMP 12 | TIMSTMP 11 | TIMSTMP 10 | TIMSTMP 9 | TIMSTMP 8 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | TIMSTMP15:8 | High byte of Time Stamp See Figure 44. | | | | | |

No default value after reset

Table 92. CANSTMPL Register
CANSTMPL (S:AEh Read Only)
CAN Stamp Timer Low

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-----------|-----------|-----------|-----------|-----------|
| TIMSTMP 7 | TIMSTMP 6 | TIMSTMP 5 | TIMSTMP 4 | TIMSTMP 3 | TIMSTMP 2 | TIMSTMP 1 | TIMSTMP 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | TIMSTMP7:0 | Low byte of Time Stamp See Figure 44. | | | | | |

No default value after reset

Table 93. CANTTCH Register
CANTTCH (S:A5h Read Only)
CAN TTC Timer High

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-----------|-----------|-----------|----------|----------|
| TIMTTC 15 | TIMTTC 14 | TIMTTC 13 | TIMTTC 12 | TIMTTC 11 | TIMTTC 10 | TIMTTC 9 | TIMTTC 8 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | TIMTTC15:8 | High byte of TTC Timer See Figure 44. | | | | | |

Reset Value = 0000 0000b

Table 94. CANTTCL Register
CANTTCL (S:A4h Read Only)
CAN TTC Timer Low

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|----------|----------|----------|----------|----------|
| TIMTTC 7 | TIMTTC 6 | TIMTTC 5 | TIMTTC 4 | TIMTTC 3 | TIMTTC 2 | TIMTTC 1 | TIMTTC 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | TIMTTC7:0 | Low Byte of TTC Timer See Figure 44. | | | | | |

Reset Value = 0000 0000b

Programmable Counter Array (PCA)

The PCA provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy. The PCA consists of a dedicated timer/counter which serves as the time base for an array of two compare/capture modules. Its clock input can be programmed to count any of the following signals:

- PCA clock frequency/6 (See “clock” section)
- PCA clock frequency/2
- Timer 0 overflow
- External input on ECI (P1.2)

Each compare/capture modules can be programmed in any one of the following modes:

- Rising and/or falling edge capture,
- Software timer
- High-speed output
- Pulse width modulator

When the compare/capture modules are programmed in capture mode, software timer, or high speed output mode, an interrupt can be generated when the module executes its function. Both modules and the PCA timer overflow share one interrupt vector.

The PCA timer/counter and compare/capture modules share Port 1 for external I/Os. These pins are listed below. If the pin is not used for the PCA, it can still be used for standard I/O.

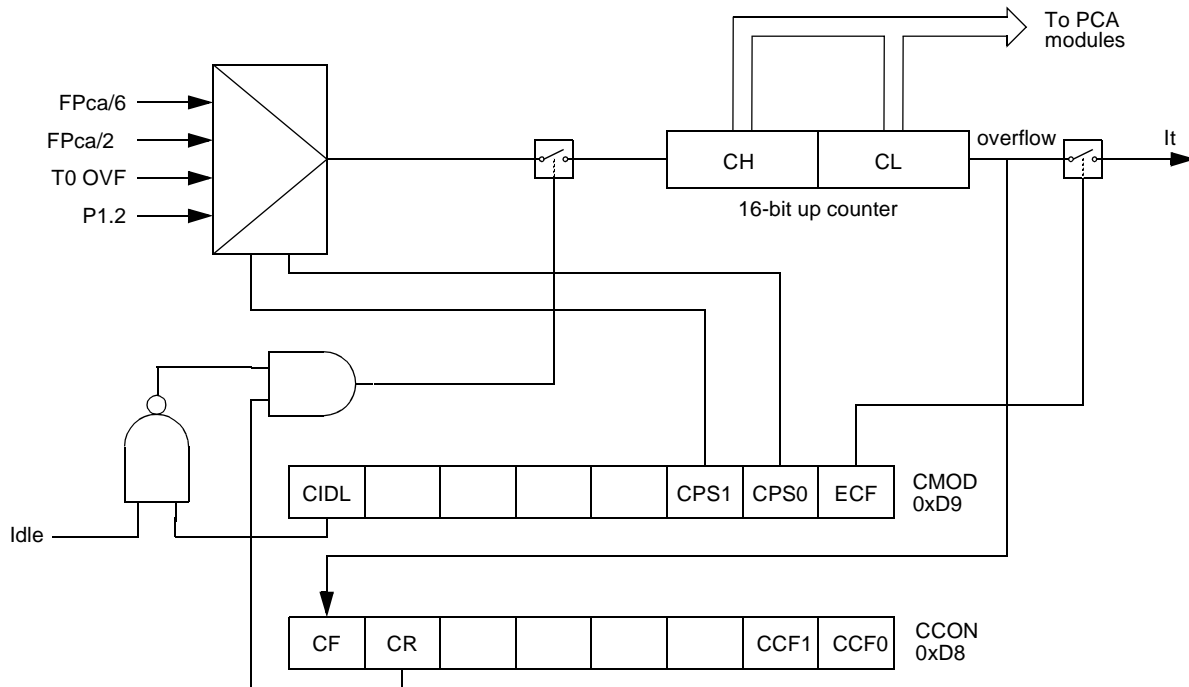
| PCA Component | External I/O Pin |
|-----------------|------------------|
| 16-bit Counter | P1.2/ECI |
| 16-bit Module 0 | P1.3/CEX0 |
| 16-bit Module 1 | P1.4/CEX1 |

PCA Timer

The PCA timer is a common time base for both modules (See Figure 9). The timer count source is determined from the CPS1 and CPS0 bits in the **CMOD SFR** (See Table 8) and can be programmed to run at:

- 1/6 the PCA clock frequency.
- 1/2 the PCA clock frequency.
- The Timer 0 overflow.
- The input on the ECI pin (P1.2).

Figure 46. PCA Timer/Counter



The CMOD register includes three additional bits associated with the PCA.

- The CIDL bit which allows the PCA to stop during idle mode.
- The ECF bit which when set causes an interrupt and the PCA overflow flag CF in CCON register to be set when the PCA timer overflows.

The CCON register contains the run control bit for the PCA and the flags for the PCA timer and each module.

- The CR bit must be set to run the PCA. The PCA is shut off by clearing this bit.
- The CF bit is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in CMOD register is set. The CF bit can only be cleared by software.
- The CCF0:1 bits are the flags for the modules (CCF0 for module0...) and are set by hardware when either a match or a capture occurs. These flags also can be cleared by software.

PCA Modules

Each one of the two compare/capture modules has six possible functions. It can perform:

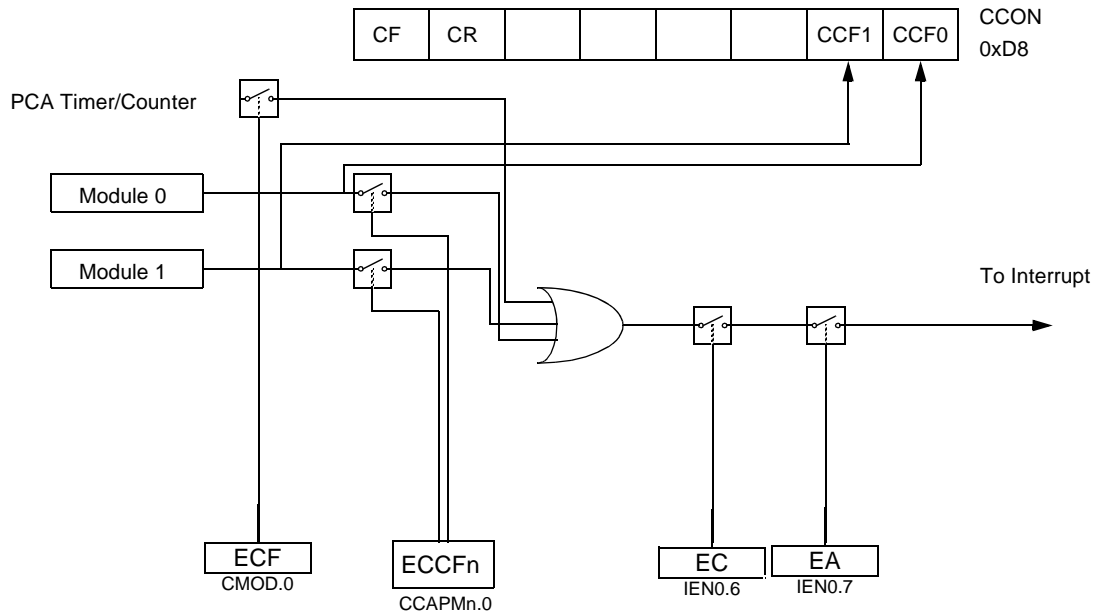
- 16-bit Capture, positive-edge triggered
- 16-bit Capture, negative-edge triggered
- 16-bit Capture, both positive and negative-edge triggered
- 16-bit Software Timer
- 16-bit High Speed Output
- 8-bit Pulse Width Modulator.

Each module in the PCA has a special function register associated with it (CCAPM0 for module 0 ...). The CCAPM0:1 registers contain the bits that control the mode that each module will operate in.

- The ECCF bit enables the CCF flag in the CCON register to generate an interrupt when a match or compare occurs in the associated module.
- The PWM bit enables the pulse width modulation mode.
- The TOG bit when set causes the CEX output associated with the module to toggle when there is a match between the PCA counter and the module's capture/compare register.
- The match bit MAT when set will cause the CCFn bit in the CCON register to be set when there is a match between the PCA counter and the module's capture/compare register.
- The two bits CAPN and CAPP in CCAPMn register determine the edge that a capture input will be active on. The CAPN bit enables the negative edge, and the CAPP bit enables the positive edge. If both bits are set both edges will be enabled.
- The bit ECOM in CCAPM register when set enables the comparator function.

PCA Interrupt

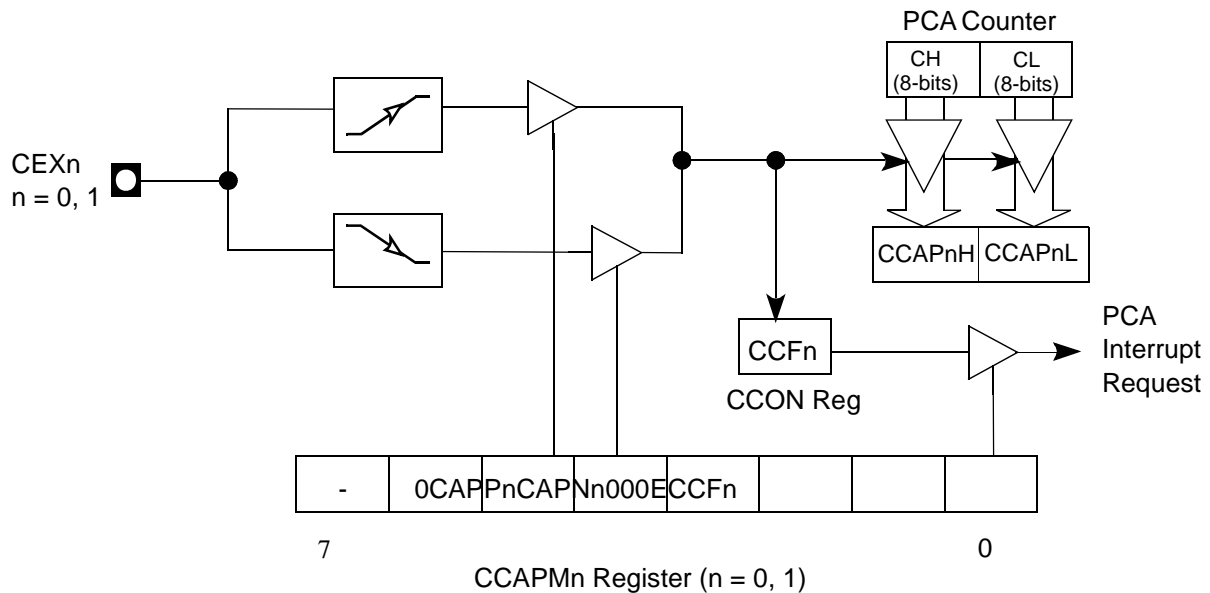
Figure 47. PCA Interrupt System



PCA Capture Mode

To use one of the PCA modules in capture mode either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set then an interrupt will be generated.

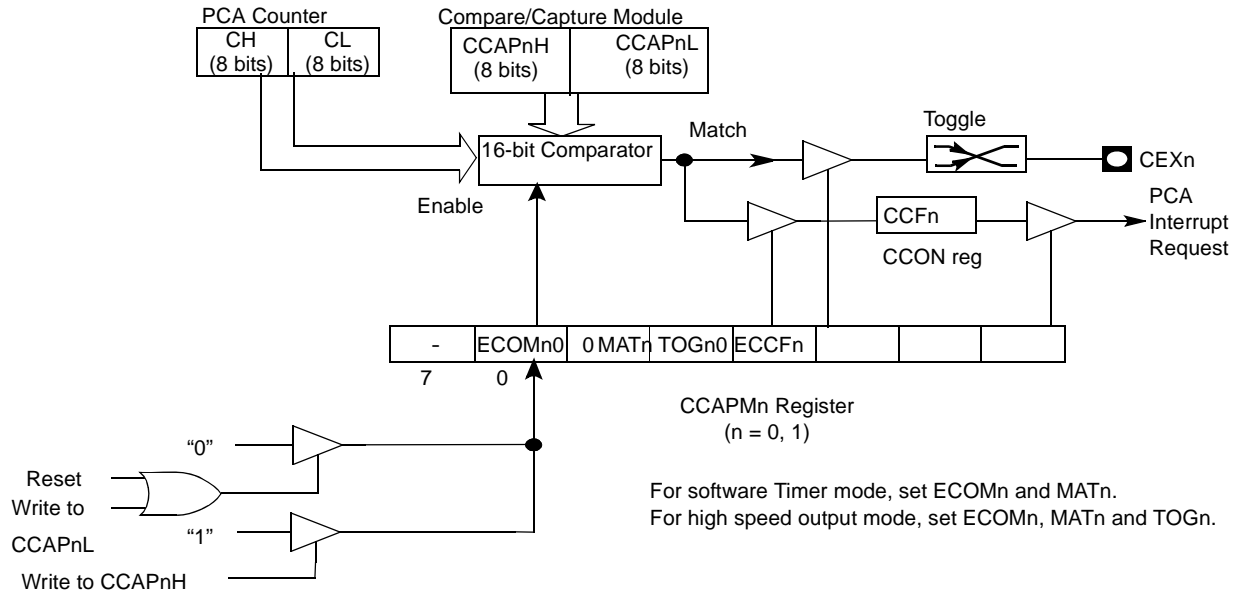
Figure 48. PCA Capture Mode



16-bit Software Timer Mode

The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers and when a match occurs an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set.

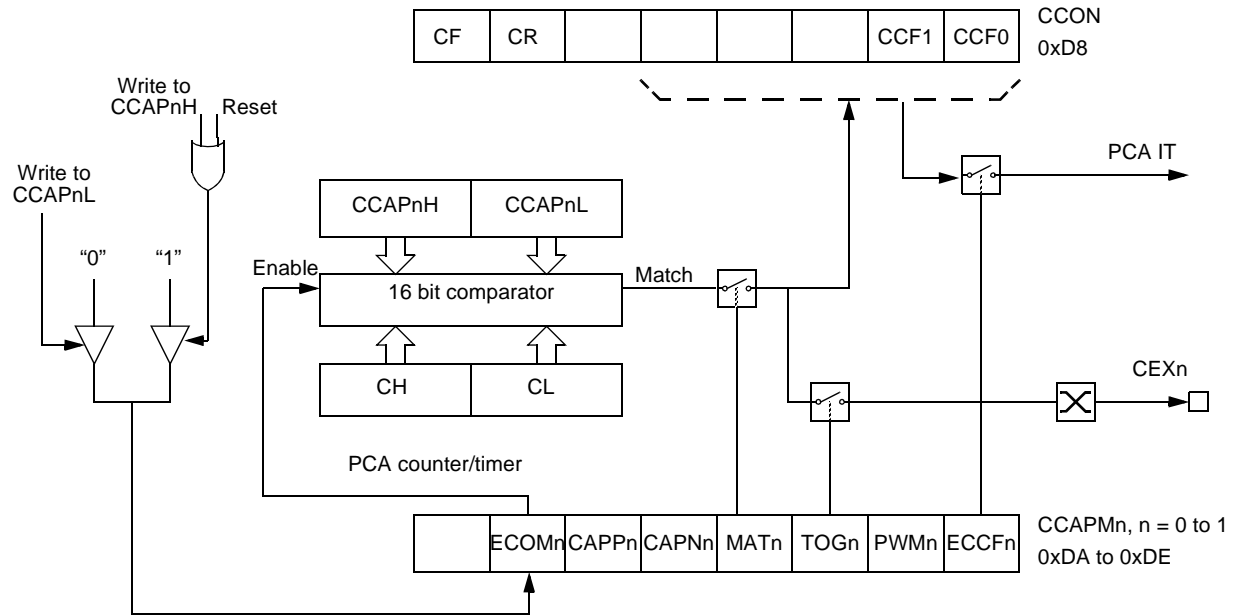
Figure 49. PCA 16-bit Software Timer and High Speed Output Mode



High Speed Output Mode

In this mode the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set.

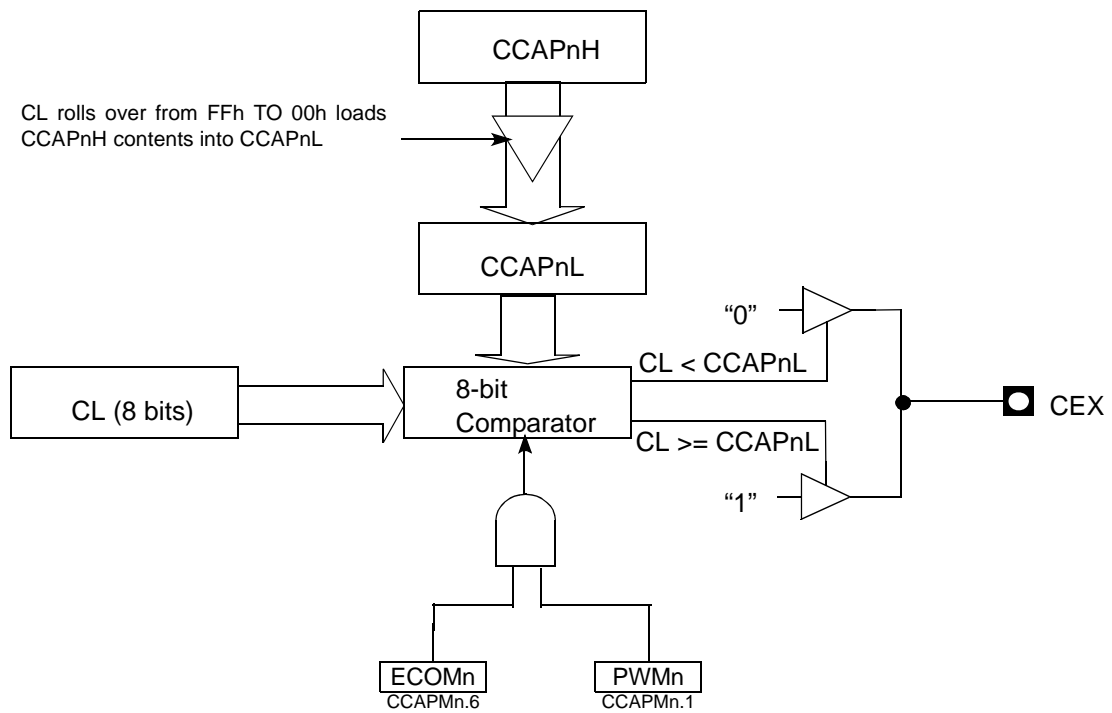
Figure 50. PCA High Speed Output Mode



Pulse Width Modulator Mode

All the PCA modules can be used as PWM outputs. The output frequency depends on the source for the PCA timer. All the modules will have the same output frequency because they all share the PCA timer. The duty cycle of each module is independently variable using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR the output will be low, when it is equal to or greater than it, the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. the allows the PWM to be updated without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

Figure 51. PCA PWM Mode



PCA Registers

Table 95. CMOD Register

CMOD (S:D9h)
PCA Counter Mode Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|------|------|-----|
| CIDL | - | - | - | - | CPS1 | CPS0 | ECF |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | CIDL | PCA Counter Idle Control bit Clear to let the PCA run during Idle mode. Set to stop the PCA when Idle mode is invoked. |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 2-1 | CPS1:0 | EWC Count Pulse Select bits <u>CPS1</u> <u>CPS0</u> <u>Clock source</u> 0 0 Internal Clock, FPca/6 0 1 Internal Clock, FPca/2 1 0 Timer 0 overflow 1 1 External clock at ECI/P1.2 pin (Max. Rate = FPca/4) |
| 0 | ECF | Enable PCA Counter Overflow Interrupt bit Clear to disable CF bit in CCON register to generate an interrupt. Set to enable CF bit in CCON register to generate an interrupt. |

Reset Value = 0XXX X000b

Table 96. CCON Register

CCON (S:D8h)
PCA Counter Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|---|---|---|---|------|------|
| CF | CR | - | - | - | - | CCF1 | CCF0 |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | CF | PCA Timer/Counter Overflow flag Set by hardware when the PCA Timer/Counter rolls over. This generates a PCA interrupt request if the ECF bit in CMOD register is set. Must be cleared by software. |
| 6 | CR | PCA Timer/Counter Run Control bit Clear to turn the PCA Timer/Counter off. Set to turn the PCA Timer/Counter on. |
| 5-2 | - | Reserved The value read from these bits are indeterminate. Do not set these bits. |
| 1 | CCF1 | PCA Module 1 Compare/Capture Flag Set by hardware when a match or capture occurs. This generates a PCA interrupt request if the ECCF 1 bit in CCAPM 1 register is set. Must be cleared by software. |
| 0 | CCF0 | PCA Module 0 Compare/Capture Flag Set by hardware when a match or capture occurs. This generates a PCA interrupt request if the ECCF 0 bit in CCAPM 0 register is set. Must be cleared by software. |

Reset Value = 00xx xx00b

Table 97. CCAPnH Registers

CCAP0H (S:FAh)

CCAP1H (S:FBh)

PCA High Byte Compare/Capture Module n Register (n=0..1)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|----------|----------|----------|----------|----------|
| CCAPnH 7 | CCAPnH 6 | CCAPnH 5 | CCAPnH 4 | CCAPnH 3 | CCAPnH 2 | CCAPnH 1 | CCAPnH 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7:0 | CCAPnH 7:0 | High byte of EWC-PCA comparison or capture values | | | | | |

Reset Value = 0000 0000b

Table 98. CCAPnL Registers

CCAP0L (S:EAh)

CCAP1L (S:EBh)

PCA Low Byte Compare/Capture Module n Register (n=0..1)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|----------|----------|----------|----------|----------|
| CCAPnL 7 | CCAPnL 6 | CCAPnL 5 | CCAPnL 4 | CCAPnL 3 | CCAPnL 2 | CCAPnL 1 | CCAPnL 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7:0 | CCAPnL 7:0 | Low byte of EWC-PCA comparison or capture values | | | | | |

Reset Value = 0000 0000b

Table 99. CCAPMn Registers

CCAPM0 (S:DAh)

CCAPM1 (S:DBh)

PCA Compare/Capture Module n Mode registers (n=0..1)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-------|------|------|------|-------|
| - | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The Value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 | ECOMn | Enable Compare Mode Module x bit Clear to disable the Compare function. Set to enable the Compare function. The Compare function is used to implement the software Timer, the high-speed output, the Pulse Width Modulator (PWM) and the Watchdog Timer (WDT). | | | | | |
| 5 | CAPPn | Capture Mode (Positive) Module x bit Clear to disable the Capture function triggered by a positive edge on CEXx pin. Set to enable the Capture function triggered by a positive edge on CEXx pin | | | | | |
| 4 | CAPNn | Capture Mode (Negative) Module x bit Clear to disable the Capture function triggered by a negative edge on CEXx pin. Set to enable the Capture function triggered by a negative edge on CEXx pin. | | | | | |
| 3 | MATn | Match Module x bit Set when a match of the PCA Counter with the Compare/Capture register sets CCFx bit in CCON register, flagging an interrupt. | | | | | |
| 2 | TOGn | Toggle Module x bit The toggle mode is configured by setting ECOMx, MATx and TOGx bits. Set when a match of the PCA Counter with the Compare/Capture register toggles the CEXx pin. | | | | | |
| 1 | PWMn | Pulse Width Modulation Module x Mode bit Set to configure the module x as an 8-bit Pulse Width Modulator with output waveform on CEXx pin. | | | | | |
| 0 | ECCFn | Enable CCFx Interrupt bit Clear to disable CCFx bit in CCON register to generate an interrupt request. Set to enable CCFx bit in CCON register to generate an interrupt request. | | | | | |

Reset Value = X000 0000b

Table 100. CH Register

CH (S:F9h)
PCA Counter Register High value

| | | | | | | | |
|------------|------|--------------|----------------------------|------|------|------|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH 7 | CH 6 | CH 5 | CH 4 | CH 3 | CH 2 | CH 1 | CH 0 |
| Bit Number | | Bit Mnemonic | Description | | | | |
| 7:0 | | CH 7:0 | High byte of Timer/Counter | | | | |

Reset Value = 0000 00000b

Table 101. CL Register

CL (S:E9h)
PCA counter Register Low value

| | | | | | | | |
|------------|------|--------------|---------------------------|------|------|------|------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CL 7 | CL 6 | CL 5 | CL 4 | CL 3 | CL 2 | CL 1 | CL 0 |
| | | | | | | | |
| Bit Number | | Bit Mnemonic | Description | | | | |
| 7:0 | | CL0 7:0 | Low byte of Timer/Counter | | | | |

Reset Value = 0000 00000b

Analog-to-Digital Converter (ADC)

This section describes the on-chip 10-bit analog-to-digital converter of the T89C51CC02. Eight ADC channels are available for sampling of the external sources AN0 to AN7. An analog multiplexer allows the single ADC converter to select one from the 8 ADC channels as ADC input voltage (ADCIN). ADCIN is converted by the 10-bit-cascaded potentiometric ADC.

Two modes of conversion are available:

- Standard conversion (8 bits).
- Precision conversion (10 bits).

For the precision conversion, set bit PSIDLE in ADCON register and start conversion. The device is in a pseudo-idle mode, the CPU does not run but the peripherals are always running. This mode allows digital noise to be as low as possible, to ensure high precision conversion.

For this mode it is necessary to work with end of conversion interrupt, which is the only way to wake the device up.

If another interrupt occurs during the precision conversion, it will be served only after this conversion is completed.

Features

- 8 channels with multiplexed inputs
- 10-bit cascaded potentiometric ADC
- Conversion time 16 micro-seconds (typ.)
- Zero Error (offset) ± 2 LSB max
- Positive External Reference Voltage Range (VAREF) 2.4 to 3.0-volt (typ.)
- ADCIN Range 0 to 3-volt
- Integral non-linearity typical 1 LSB, max. 2 LSB
- Differential non-linearity typical 0.5 LSB, max. 1 LSB
- Conversion Complete Flag or Conversion Complete Interrupt
- Selectable ADC Clock

ADC Port1 I/O Functions

Port 1 pins are general I/O that are shared with the ADC channels. The channel select bit in ADCF register define which ADC channel/port1 pin will be used as ADCIN. The remaining ADC channels/port1 pins can be used as general purpose I/O or as the alternate function that is available.

A conversion launched on a channel which are not selected on ADCF register will not have any effect.

VAREF

VAREF should be connected to a low impedance point and must remain in the range specified VAREF absolute maximum range (See section "AC-DC").

. If the ADC is not used, it is recommended to tie VAREF to VAGND.

Figure 52. ADC Description

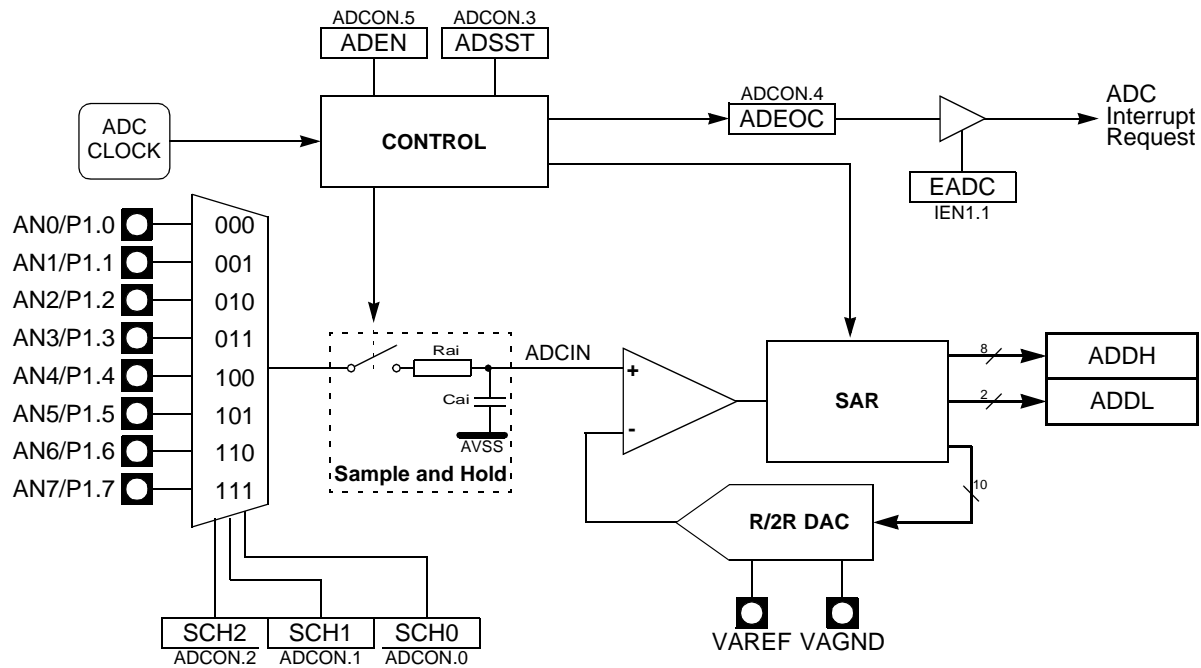
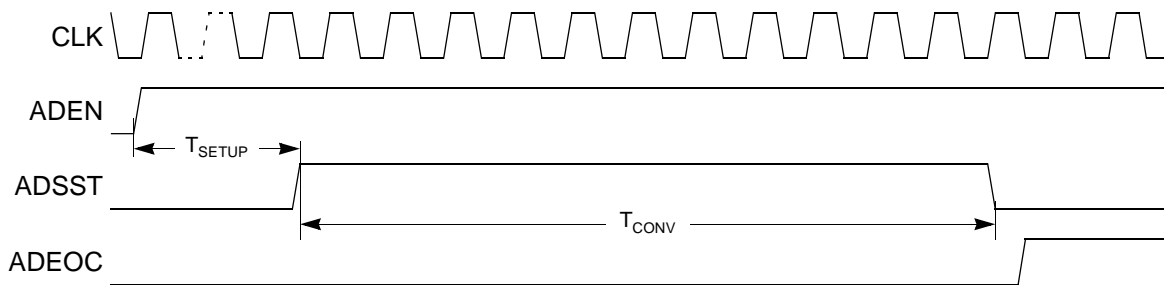


Figure 53 shows the timing diagram of a complete conversion. For simplicity, the figure depicts the waveforms in idealized form and do not provide precise timing information. For ADC characteristics and timing parameters refer to the section “AC Characteristics” of this datasheet.

Figure 53. Timing Diagram



Note: Tsetup min, see the AC Parameter for A/D conversion.
Tconv = 11 clock ADC = 1sample and hold + 10-bit conversion
The user must ensure that Tsetup time between setting ADEN and the start of the first conversion.

ADC Converter Operation

A start of single A/D conversion is triggered by setting bit ADSST (ADCON.3).

After completion of the A/D conversion, the ADSST bit is cleared by hardware.

The end-of-conversion flag ADEOC (ADCON.4) is set when the value of conversion is available in ADDH and ADDL, it must be cleared by software. If the bit EADC (IEN1.1) is set, an interrupt occur when flag ADEOC is set (See Figure 55). Clear this flag for re-arming the interrupt.

Note: Always leave Tsetup time before starting a conversion unless ADEN is permanently high. In this case one should wait Tsetup only before the first conversion

The bits SCH0 to SCH2 in ADCON register are used for the analog input channel selection.

Table 102. Selected Analog input

| SCH2 | SCH1 | SCH0 | Selected Analog Input |
|------|------|------|-----------------------|
| 0 | 0 | 0 | AN0 |
| 0 | 0 | 1 | AN1 |
| 0 | 1 | 0 | AN2 |
| 0 | 1 | 1 | AN3 |
| 1 | 0 | 0 | AN4 |
| 1 | 0 | 1 | AN5 |
| 1 | 1 | 0 | AN6 |
| 1 | 1 | 1 | AN7 |

Voltage Conversion

When the ADCIN is equals to VAREF the ADC converts the signal to 3FFh (full scale). If the input voltage equals VAGND, the ADC converts it to 000h. Input voltage between VAREF and VAGND are a straight-line linear conversion. All other voltages will result in 3FFh if greater than VAREF and 000h if less than VAGND.

Note that ADCIN should not exceed VAREF absolute maximum range (See section “AC-DC”).

Clock Selection

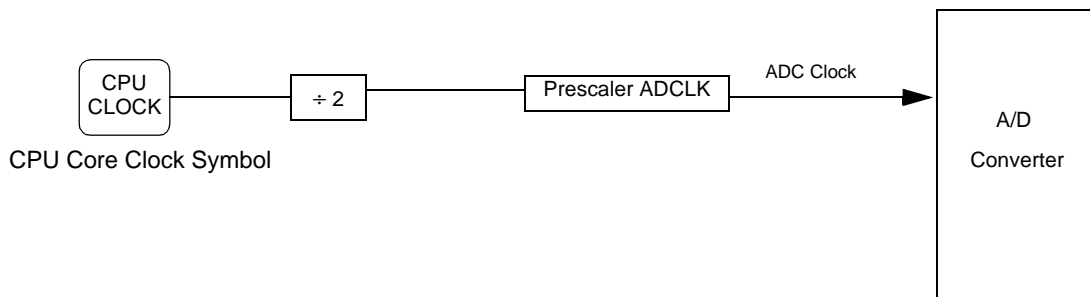
The ADC clock is the same as CPU.

The maximum clock frequency is defined in the DC parameter for A/D converter. A prescaler is featured (ADCCLK) to generate the ADC clock from the oscillator frequency.

if $PRS = 0$ then $F_{ADC} = F_{periph} / 64$

if $PRS > 0$ then $F_{ADC} = F_{periph} / 2 \times PRS$

Figure 54. A/D Converter Clock



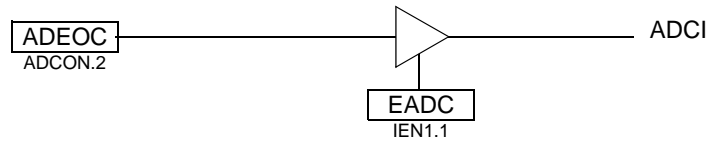
ADC Standby Mode

When the ADC is not used, it is possible to set it in standby mode by clearing bit ADEN in ADCON register. In this mode the power dissipation is reduced.

IT ADC Management

An interrupt end-of-conversion will occur when the bit ADEOC is activated and the bit EADC is set. For re-arming the interrupt the bit ADEOC must be cleared by software.

Figure 55. ADC interrupt structure



Routine Examples

```

1. Configure P1.2 and P1.3 in ADC channels
// configure channel P1.2 and P1.3 for ADC
ADCF = 0Ch

// Enable the ADC
ADCON = 20h

2. Start a standard conversion
// The variable 'channel' contains the channel to convert
// The variable 'value_converted' is an unsigned int
// Clear the field SCH[2:0]
ADCON &= F8h
// Select channel
ADCON |= channel
// Start conversion in standard mode
ADCON |= 08h
// Wait flag End of conversion
while((ADCON & 01h) != 01h)
// Clear the End of conversion flag
ADCON &= EFh
// read the value
value_converted = (ADDH << 2) + (ADDL)

3. Start a precision conversion (need interrupt ADC)
// The variable 'channel' contains the channel to convert
// Enable ADC
EADC = 1
// clear the field SCH[2:0]
ADCON &= F8h
// Select the channel
ADCON |= channel
// Start conversion in precision mode
ADCON |= 48h
  
```

Note: To enable the ADC interrupt: EA = 1

Registers

Table 103. ADCF Register
ADCF (S:F6h)
ADC Configuration

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|------|------|------|------|------|
| CH 7 | CH 6 | CH 5 | CH 4 | CH 3 | CH 2 | CH 1 | CH 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | CH 0:7 | Channel Configuration Set to use P1.x as ADC input. Clear to use P1.x as standart I/O port. | | | | | |

Reset Value = 0000 0000b

Table 104. ADCON Register
ADCON (S:F3h)
ADC Control Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|-------|-------|------|------|------|
| - | PSIDLE | ADEN | ADEOC | ADSST | SCH2 | SCH1 | SCH0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from these bits are indeterminate. Do not set these bits. | | | | | |
| 6 | PSIDLE | Pseudo Idle Mode (Best Precision) Set to put in idle mode during conversion Clear to convert without idle mode. | | | | | |
| 5 | ADEN | Enable/Standby Mode Set to enable ADC Clear for Standby mode. | | | | | |
| 4 | ADEOC | End Of Conversion Set by hardware when ADC result is ready to be read. This flag can generate an interrupt. Must be cleared by software. | | | | | |
| 3 | ADSST | Start and Status Set to start an A/D conversion. Cleared by hardware after completion of the conversion | | | | | |
| 2-0 | SCH2:0 | Selection of Channel to Convert See Table 102 | | | | | |

Reset Value = X000 0000b

Table 105. ADCLK Register
ADCLK (S:F2h)
ADC Clock Prescaler

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-------|-------|-------|-------|-------|
| - | - | - | PRS 4 | PRS 3 | PRS 2 | PRS 1 | PRS 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 5 | - | Reserved The value read from these bits are indeterminate. Do not set these bits. | | | | | |
| 4-0 | PRS4:0 | Clock Prescaler $F_{adc} = F_{cpuclock}/(4*PRS)$ in X1 mode $F_{adc} = F_{cpuclock}/(2*PRS)$ in X2 mode | | | | | |

Reset Value = XXX0 0000b

Table 106. ADDH Register
ADDH (S:F5h Read Only)
ADC Data High Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|-------------------------------|--------|--------|--------|--------|--------|
| ADAT 9 | ADAT 8 | ADAT 7 | ADAT 6 | ADAT 5 | ADAT 4 | ADAT 3 | ADAT 2 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 0 | ADAT9:2 | ADC result bits 9-2 | | | | | |

Reset Value = 00h

Table 107. ADDL Register
ADDL (S:F4h Read Only)
ADC Data Low Byte Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|---|---|---|--------|--------|
| - | - | - | - | - | - | ADAT 1 | ADAT 0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 - 2 | - | Reserved The value read from these bits are indeterminate. Do not set these bits. | | | | | |
| 1-0 | ADAT1:0 | ADC result bits 1-0 | | | | | |

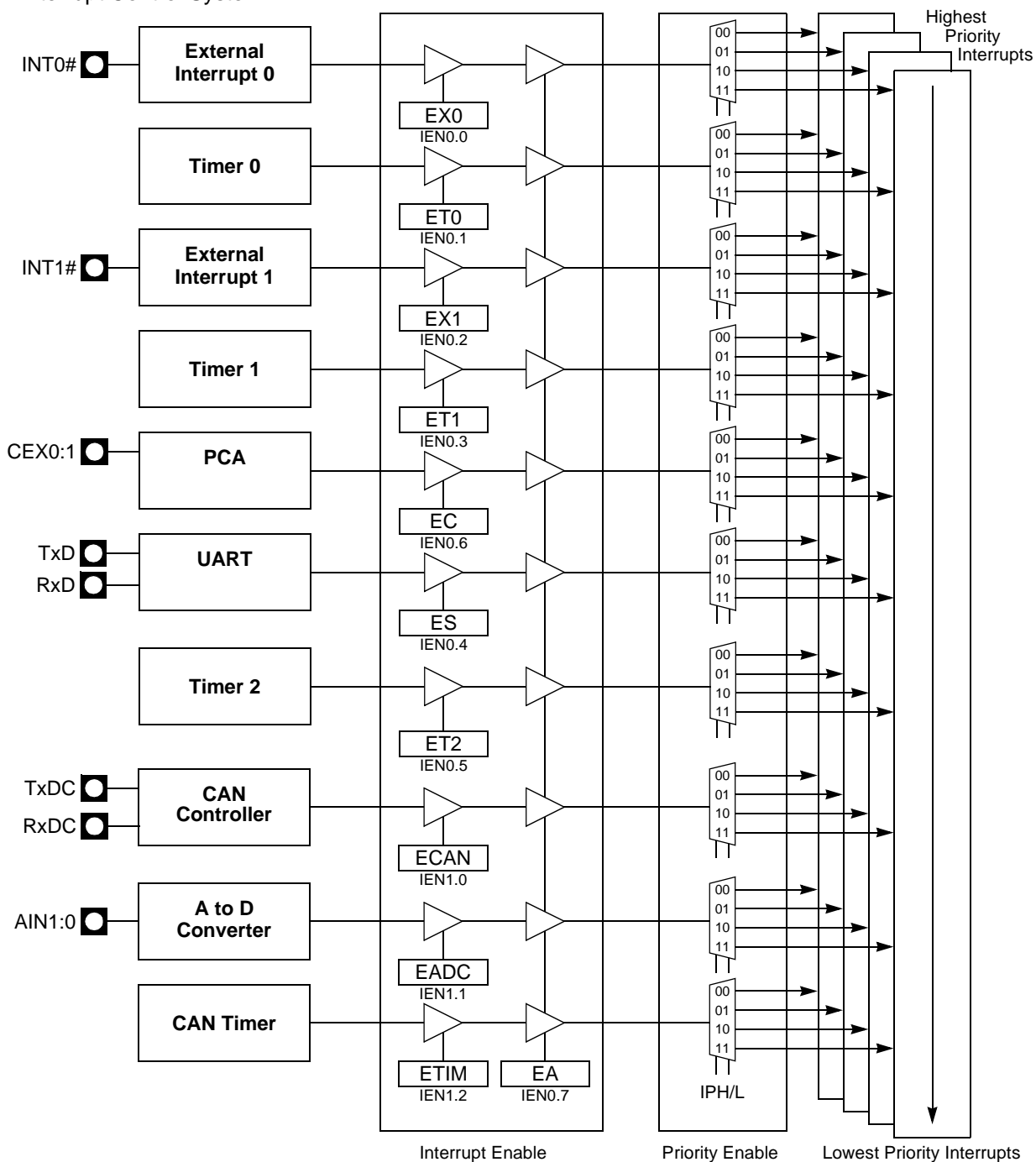
Reset Value = 00h

Interrupt System

Introduction

The CAN Controller has a total of 10 interrupt vectors: two external interrupts ($\overline{\text{INT0}}$ and $\overline{\text{INT1}}$), three timer interrupts (timers 0, 1 and 2), a serial port interrupt, a PCA, a CAN interrupt, a timer overrun interrupt and an ADC. These interrupts are shown below.

Figure 56. Interrupt Control System



Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the Interrupt Enable register. This register also contains a global disable bit which must be cleared to disable all the interrupts at the same time.

Each interrupt source can also be individually programmed to one of four priority levels by setting or clearing a bit in the Interrupt Priority registers. The Table below shows the bit values and priority levels associated with each combination.

Table 108. Priority Level bit Values

| IPH.x | IPL.x | Interrupt Level Priority |
|-------|-------|--------------------------|
| 0 | 0 | 0 (Lowest) |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 (Highest) |

A low-priority interrupt can be interrupted by a high priority interrupt but not by another low-priority interrupt. A high-priority interrupt cannot be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of the higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence, See Table 109.

Table 109. Interrupt Priority Within Level

| Interrupt Name | Interrupt Address Vector | Interrupt Number | Polling Priority |
|---------------------------------|--------------------------|------------------|------------------|
| External interrupt (INT0) | 0003h | 1 | 1 |
| Timer0 (TF0) | 000Bh | 2 | 2 |
| External interrupt (INT1) | 0013h | 3 | 3 |
| Timer 1 (TF1) | 001Bh | 4 | 4 |
| PCA (CF or CCFn) | 0033h | 7 | 5 |
| UART (RI or TI) | 0023h | 5 | 6 |
| Timer 2 (TF2) | 002Bh | 6 | 7 |
| CAN (Txok, Rxok, Err or OvrBuf) | 003Bh | 8 | 8 |
| ADC (ADCI) | 0043h | 9 | 9 |
| CAN Timer Overflow (OVRTIM) | 004Bh | 10 | 10 |

Registers

Figure 57. IEN0 Register
IEN0 (S:A8h)
Interrupt Enable Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|---|----|-----|-----|-----|-----|
| EA | EC | ET2 | ES | ET1 | EX1 | ET0 | EX0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | EA | Enable All Interrupt bit Clear to disable all interrupts. Set to enable all interrupts. If EA=1, each interrupt source is individually enabled or disabled by setting or clearing its interrupt enable bit. | | | | | |
| 6 | EC | PCA Interrupt Enable Clear to disable the PCA interrupt. Set to enable the PCA interrupt. | | | | | |
| 5 | ET2 | Timer 2 Overflow Interrupt Enable bit Clear to disable Timer 2 overflow interrupt. Set to enable Timer 2 overflow interrupt. | | | | | |
| 4 | ES | Serial port Enable bit Clear to disable serial port interrupt. Set to enable serial port interrupt. | | | | | |
| 3 | ET1 | Timer 1 Overflow Interrupt Enable bit Clear to disable timer 1 overflow interrupt. Set to enable timer 1 overflow interrupt. | | | | | |
| 2 | EX1 | External Interrupt 1 Enable bit Clear to disable external interrupt 1. Set to enable external interrupt 1. | | | | | |
| 1 | ET0 | Timer 0 Overflow Interrupt Enable bit Clear to disable timer 0 overflow interrupt. Set to enable timer 0 overflow interrupt. | | | | | |
| 0 | EX0 | External Interrupt 0 Enable bit Clear to disable external interrupt 0. Set to enable external interrupt 0. | | | | | |

Reset Value = 0000 0000b
bit addressable

Figure 58. IEN1 Register
 IEN1 (S:E8h)
 Interrupt Enable Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|---|---|------|------|------|
| - | - | - | - | | ETIM | EADC | ECAN |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 2 | ETIM | Timer overrun Interrupt Enable bit Clear to disable the timer overrun interrupt. Set to enable the timer overrun interrupt. | | | | | |
| 1 | EADC | ADC Interrupt Enable bit Clear to disable the ADC interrupt. Set to enable the ADC interrupt. | | | | | |
| 0 | ECAN | CAN Interrupt Enable bit Clear to disable the CAN interrupt. Set to enable the CAN interrupt. | | | | | |

Reset Value = xxxx x000b
 bit addressable

Table 110. IPL0 Register
IPL0 (S:B8h)
Interrupt Enable Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|----|-----|-----|-----|-----|
| - | PPC | PT2 | PS | PT1 | PX1 | PT0 | PX0 |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 | PPC | PCA Interrupt Priority bit Refer to PPCH for priority level. | | | | | |
| 5 | PT2 | Timer 2 Overflow Interrupt Priority bit Refer to PT2H for priority level. | | | | | |
| 4 | PS | Serial Port Priority bit Refer to PSH for priority level. | | | | | |
| 3 | PT1 | Timer 1 Overflow Interrupt Priority bit Refer to PT1H for priority level. | | | | | |
| 2 | PX1 | External Interrupt 1 Priority bit Refer to PX1H for priority level. | | | | | |
| 1 | PT0 | Timer 0 Overflow Interrupt Priority bit Refer to PT0H for priority level. | | | | | |
| 0 | PX0 | External Interrupt 0 Priority bit Refer to PX0H for priority level. | | | | | |

Reset Value = X000 0000b
bit addressable

Table 111. IPL1 Register
IPL1 (S:F8h)
Interrupt Priority Low Register 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|-------|-------|-------|
| - | - | - | - | | POVRL | PADCL | PCANL |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 2 | POVRL | Timer Overrun Interrupt Priority Level Less Significant bit Refer to PI2CH for priority level. |
| 1 | PADCL | ADC Interrupt Priority Level Less Significant bit Refer to PSPIH for priority level. |
| 0 | PCANL | CAN Interrupt Priority Level Less Significant bit Refer to PKBH for priority level. |

Reset Value = XXXX X000b
bit addressable

Table 112. IPH0 Register
IPH0 (B7h)
Interrupt High Priority Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|--------------|--|-----|------|------|------|------|
| - | PPCH | PT2H | PSH | PT1H | PX1H | PT0H | PX0H |
| Bit Number | Bit Mnemonic | Description | | | | | |
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. | | | | | |
| 6 | PPCH | PCA Interrupt Priority Level Most Significant bit <u>PPCH</u> <u>PPC</u> <u>Priority level</u> 0 0 Lowest 0 1 1 0 1 1 Highest priority | | | | | |
| 5 | PT2H | Timer 2 Overflow Interrupt High Priority bit <u>PT2H</u> <u>PT2</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |
| 4 | PSH | Serial Port High Priority bit <u>PSH</u> <u>PS</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |
| 3 | PT1H | Timer 1 Overflow Interrupt High Priority bit <u>PT1H</u> <u>PT1</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |
| 2 | PX1H | External Interrupt 1 High Priority bit <u>PX1H</u> <u>PX1</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |
| 1 | PT0H | Timer 0 Overflow Interrupt High Priority bit <u>PT0H</u> <u>PT0</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |
| 0 | PX0H | External Interrupt 0 High Priority bit <u>PX0H</u> <u>PX0</u> <u>Priority Level</u> 0 0 Lowest 0 1 1 0 1 1 Highest | | | | | |

Reset Value = X000 0000b

Table 113. IPH1 Register
 IPH1 (S:F7h)
 Interrupt high priority Register 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|-------|-------|-------|
| - | - | - | - | | POVRH | PADCH | PCANH |

| Bit Number | Bit Mnemonic | Description |
|------------|--------------|--|
| 7 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 6 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 5 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 4 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 3 | - | Reserved The value read from this bit is indeterminate. Do not set this bit. |
| 2 | POVRH | Timer Overrun Interrupt Priority Level Most Significant bit <u>POVRH</u> <u>POVRL</u> <u>Priority level</u> 0 0 Lowest 0 1 1 0 1 1 Highest |
| 1 | PADCH | ADC Interrupt Priority Level Most Significant bit <u>PADCH</u> <u>PADCL</u> <u>Priority level</u> 0 0 Lowest 0 1 1 0 1 1 Highest |
| 0 | PCANH | CAN Interrupt Priority Level Most Significant bit <u>PCANH</u> <u>PCANL</u> <u>Priority level</u> 0 0 Lowest 0 1 1 0 1 1 Highest |

Reset Value = XXXX X000b

Electrical Characteristics

Absolute Maximum Ratings

| | |
|---|--------------------------|
| I = industrial | -40°C to 85°C |
| Storage Temperature | -65°C to + 150°C |
| Voltage on V_{CC} from V_{SS} | -0.5V to + 6V |
| Voltage on Any Pin from V_{SS} | -0.5V to $V_{CC} + 0.2V$ |
| Power Dissipation | 1 W |

Note: Stresses at or above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.
Power Dissipation value is based on the maximum allowable die temperature and the thermal resistance of the package.

DC Parameters for Standard Voltage

$T_A = -40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$; $V_{SS} = 0\text{ V}$; $V_{CC} = 3\text{ volts}$ to 5.5 volts ; $F = 0$ to 40 MHz

Table 114. DC Parameters in Standard Voltage

| Symbol | Parameter | Min | Typ ⁽¹⁾ | Max | Unit | Test Conditions |
|-----------------|--|--|--------------------|--------------------|---------------|---|
| V_{IL} | Input Low Voltage | -0.5 | | $0.2V_{CC} - 0.1$ | V | |
| V_{IH} | Input High Voltage except XTAL1, RST | $0.2 V_{CC} + 0.9$ | | $V_{CC} + 0.5$ | V | |
| $V_{IH1}^{(2)}$ | Input High Voltage, XTAL1, RST | $0.7 V_{CC}$ | | $V_{CC} + 0.5$ | V | |
| V_{OL} | Output Low Voltage, ports 1, 2, 3 and 4 ⁽³⁾ | | | 0.3 0.45 1.0 | V V V | $I_{OL} = 100\text{ }\mu\text{A}$ $I_{OL} = 1.6\text{ mA}$ $I_{OL} = 3.5\text{ mA}$ |
| V_{OH} | Output High Voltage, ports 1, 2, 3, 4 and 5 | $V_{CC} - 0.3$ $V_{CC} - 0.7$ $V_{CC} - 1.5$ | | | V V V | $I_{OH} = -10\text{ }\mu\text{A}$ $I_{OH} = -30\text{ }\mu\text{A}$ $I_{OH} = -60\text{ }\mu\text{A}$ $V_{CC} = 5V \pm 10\%$ |
| R_{RST} | RST Pulldown Resistor | 50 | 90 | 200 | k Ω | |
| I_{IL} | Logical 0 Input Current ports 1, 2, 3 and 4 | | | -50 | μA | $V_{in} = 0.45V$ |
| I_{LI} | Input Leakage Current | | | ± 10 | μA | $0.45V < V_{in} < V_{CC}$ |
| I_{TL} | Logical 1 to 0 Transition Current, ports 1, 2, 3 and 4 | | | -650 | μA | $V_{in} = 2.0V$ |
| C_{IO} | Capacitance of I/O Buffer | | | 10 | pF | $F_c = 1\text{ MHz}$ $T_A = 25^{\circ}\text{C}$ |
| I_{PD} | Power-down Current | | 160 | 400 | μA | $3V < V_{CC} < 5.5V^{(4)}$ |
| I_{CC} | Power Supply Current $I_{CCOP}^{(6)} = 0.7\text{ Freq (MHz)} + 3\text{ mA}$ $I_{CCIDLE}^{(5)} = 0.6\text{ Freq (MHz)} + 2\text{ mA}$ | | | | | |

Notes: 1. Typicals are based on a limited number of samples and are not guaranteed. The values listed are at room temperature.
2. Flash retention is guaranteed with the same formula for V_{CC} min down to 0V.
3. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin: 10 mA

Maximum I_{OL} per 8-bit port:

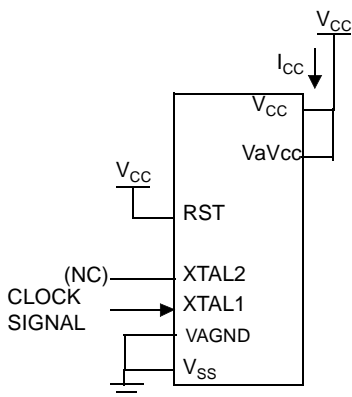
Ports 1, 2 and 3: 15 mA

Maximum total I_{OL} for all output pins: 71 mA

If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

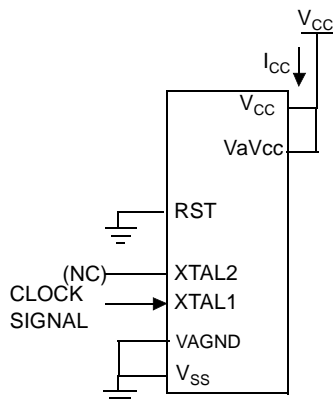
4. Power-down I_{CC} is measured with all output pins disconnected; XTAL2 N.C.; RST = V_{SS} (See Figure 61.).
5. Idle I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH} , T_{CHCL} = 5 ns, V_{IL} = $V_{SS} + 0.5V$, V_{IH} = $V_{CC} - 0.5V$; XTAL2 N.C.; RST = V_{SS} (See Figure 60.).
6. Operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with T_{CLCH} , T_{CHCL} = 5 ns (See Figure 62.), V_{IL} = $V_{SS} + 0.5V$, V_{IH} = $V_{CC} - 0.5V$; XTAL2 N.C.; RST = V_{CC} . I_{CC} would be slightly higher if a crystal oscillator used (See Figure 59.).

Figure 59. I_{CC} Test Condition, Active Mode



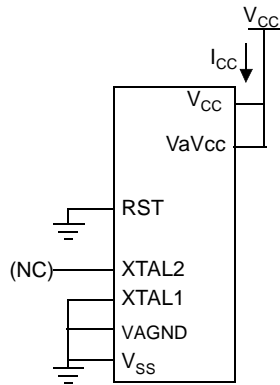
All other pins are disconnected.

Figure 60. I_{CC} Test Condition, Idle Mode



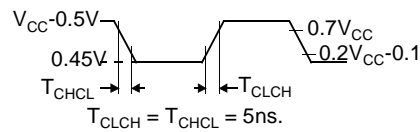
All other pins are disconnected

Figure 61. I_{CC} Test Condition, Power-down Mode



All other pins are disconnected.

Figure 62. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes



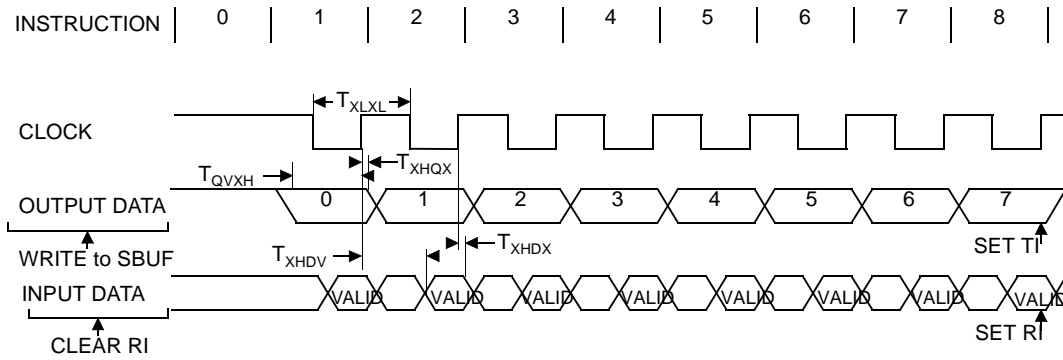
DC Parameters for A/D Converter

Table 115. DC Parameters for AD Converter in Precision Conversion

| Symbol | Parameter | Min | Typ ⁽¹⁾ | Max | Unit | Test Conditions |
|---------------------|---------------------------------|----------|--------------------|----------------|------|-----------------|
| AVin | Analog input voltage | Vss- 0.2 | | Max Vref + 0.6 | V | |
| VaVcc | Analog supply voltage | Vref | Vcc | Vcc + 10% | V | |
| Rref ⁽²⁾ | Resistance between Vref and Vss | 12 | 16 | 24 | KΩ | |
| Varef | Reference voltage | 2.40 | | 3.00 | V | |
| Cai | Analog input Capacitance | | 60 | | pF | During sampling |
| Rai | Analog input Resistor | | | 400 | Ω | During sampling |
| INL | Integral non linearity | | 1 | 2 | lsb | |
| DNL | Differential non linearity | | 0.5 | 1 | lsb | |
| OE | Offset error | -2 | | 2 | lsb | |

Notes: 1. Typicals are based on a limited number of samples and are not guaranteed.
2. With ADC enabled.

Shift Register Timing Waveforms

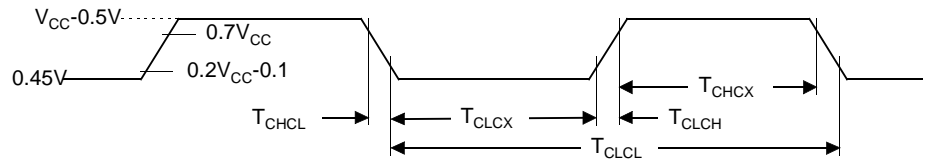


External Clock Drive Characteristics (XTAL1)

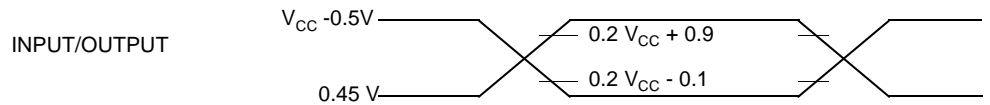
Table 119. AC Parameters

| Symbol | Parameter | Min | Max | Units |
|---------------------|-------------------------|-----|-----|-------|
| T_{CLCL} | Oscillator Period | 25 | | ns |
| T_{CHCX} | High Time | 5 | | ns |
| T_{CLCX} | Low Time | 5 | | ns |
| T_{CLCH} | Rise Time | | 5 | ns |
| T_{CHCL} | Fall Time | | 5 | ns |
| T_{CHCX}/T_{CLCX} | Cyclic ratio in X2 Mode | 40 | 60 | % |

External Clock Drive Waveforms

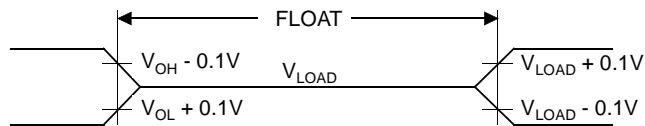


AC Testing Input/Output Waveforms



AC inputs during testing are driven at $V_{CC} - 0.5$ for a logic "1" and $0.45V$ for a logic "0". Timing measurement are made at V_{IH} min for a logic "1" and V_{IL} max for a logic "0".

Float Waveforms



For timing purposes as port pin is no longer floating when a 100 mV change from load voltage occurs and begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OL}/I_{OH} \geq \pm 20\text{mA}$.

Clock Waveforms

Valid in normal clock mode. In X2 Mode XTAL2 must be changed to XTAL2/2.

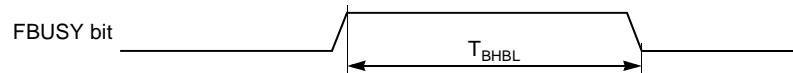
Flash/EEPROM Memory

Table 120. Memory AC Timing

$V_{CC} = 3.0\text{V}$ to 5.5V , $T_A = -40^\circ\text{C}$ to $+85^\circ\text{C}$

| Symbol | Parameter | Min | Typ | Max | Unit |
|------------|---|---------|-----|-----|--------|
| T_{BHBL} | Flash/EEPROM Internal Busy (Programming) Time | | 13 | 17 | ms |
| N_{FCY} | Number of Flash/EEPROM Erase/Write Cycles | 100 000 | | | cycles |
| T_{FDR} | Flash/EEPROM Data Retention Time | 10 | | | years |

Figure 63. Flash Memory - Internal Busy Waveforms



A/D Converter

Table 121. AC Parameters for A/D Conversion

| Symbol | Parameter | Min | Typ | Max | Unit |
|---------------------|-----------|-----|-----|-----|---------------|
| T_{SETUP} | | 4 | | | μs |
| ADC Clock Frequency | | | 700 | | KHz |

Ordering Information

| Part Number | Bootloader | Temperature Range | Package | Packing | Product Marking |
|---------------------|---------------------|--------------------|----------|----------|-----------------|
| T89C51CC02CA-RATIM | OBSOLETE | OBSOLETE | OBSOLETE | OBSOLETE | OBSOLETE |
| T89C51CC02CA-SISIM | | | | | |
| T89C51CC02CA-TDSIM | | | | | |
| T89C51CC02CA-TISIM | | | | | |
| T89C51CC02UA-RATIM | | | | | |
| T89C51CC02UA-SISIM | | | | | |
| T89C51CC02UA-TDSIM | | | | | |
| T89C51CC02UA-TISIM | | | | | |
| AT89C51CC02CA-RATUM | CAN ⁽²⁾ | Industrial & Green | VQFP32 | Tray | 89C51CC02CA-UM |
| AT89C51CC02CA-SISUM | CAN ⁽²⁾ | Industrial & Green | PLCC28 | Stick | 89C51CC02CA-UM |
| AT89C51CC02CA-TDSUM | CAN ⁽²⁾ | Industrial & Green | SOIC24 | Stick | 89C51CC02CA-UM |
| AT89C51CC02CA-TISUM | CAN ⁽²⁾ | Industrial & Green | SOIC28 | Stick | 89C51CC02CA-UM |
| AT89C51CC02UA-RATUM | UART ⁽²⁾ | Industrial & Green | VQFP32 | Tray | 89C51CC02UA-UM |
| AT89C51CC02UA-SISUM | UART ⁽²⁾ | Industrial & Green | PLCC28 | Stick | 89C51CC02UA-UM |
| AT89C51CC02UA-TDSUM | UART ⁽²⁾ | Industrial & Green | SOIC24 | Stick | 89C51CC02UA-UM |
| AT89C51CC02UA-TISUM | UART ⁽²⁾ | Industrial & Green | SOIC28 | Stick | 89C51CC02UA-UM |

Factory default programming for T89C51CC02CA-xxxx is Bootloader CAN and HSB = BBh:

- X1 mode
- BLJB = 0 : jump to Bootloader
- LB2 = 0 : Security Level 3.⁽¹⁾

Factory default programming for T89C51CC02UA-xxxx is Bootloader UART and HSB = BBh:

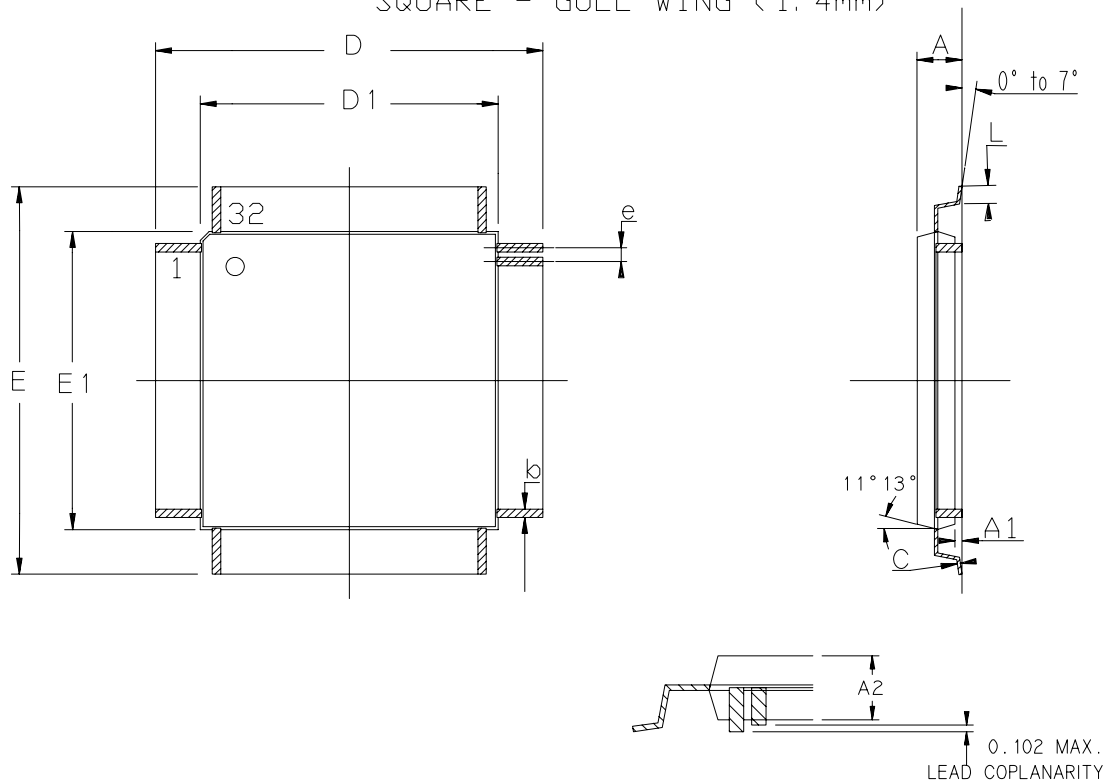
- X1 mode
- BLJB = 0 : jump to Bootloader
- LB2 = 0 : Security Level 3.⁽¹⁾

Notes: 1. LB2 = 0 is not described in Table 22 Program load bit. LB2 = 0 is equivalent to LB1 = 0: Security Level 3.
2. Customer can change these modes by re-programming with a parallel programmer, this can be done by an Atmel distributor.

Package Drawings

VQFP32

32L SHRINK QUAD FLAT PACK
SQUARE - GULL WING < 1,4mm



| | MM | | INCH | |
|----|----------|------|-----------|------|
| | Min | Max | Min | Max |
| A | - | 1.60 | - | .063 |
| A1 | 0.05 | 0.15 | .002 | .006 |
| A2 | 1.35 | 1.45 | .053 | .057 |
| C | 0.09 | 0.20 | .004 | .008 |
| D | 9.00 BSC | | .354 BSC | |
| D1 | 7.00 BSC | | .276 BSC | |
| E | 9.00 BSC | | .354 BSC | |
| E1 | 7.00 BSC | | .276 BSC | |
| L | 0.45 | 0.75 | .018 | .030 |
| e | 0.80 BSC | | .0315 BSC | |
| b | 0.30 | 0.45 | .012 | .018 |

STANDARD NOTES FOR PQFP / VQFP / TQFP / DQFP

1/ CONTROLLING DIMENSIONS : INCHES

2/ ALL DIMENSIONING AND TOLERANCING CONFORM TO ANSI Y 14.5M - 1982.

**3/ "D1 AND E1" DIMENSIONS DO NOT INCLUDE MOLD PROTUSIONS.
MOLD PROTUSIONS SHALL NOT EXCEED 0.25 mm (0.010 INCH).
THE TOP PACKAGE BODY SIZE MAY BE SMALLER THAN THE BOTTOM
PACKAGE BODY SIZE BY AS MUCH AS 0.15 mm.**

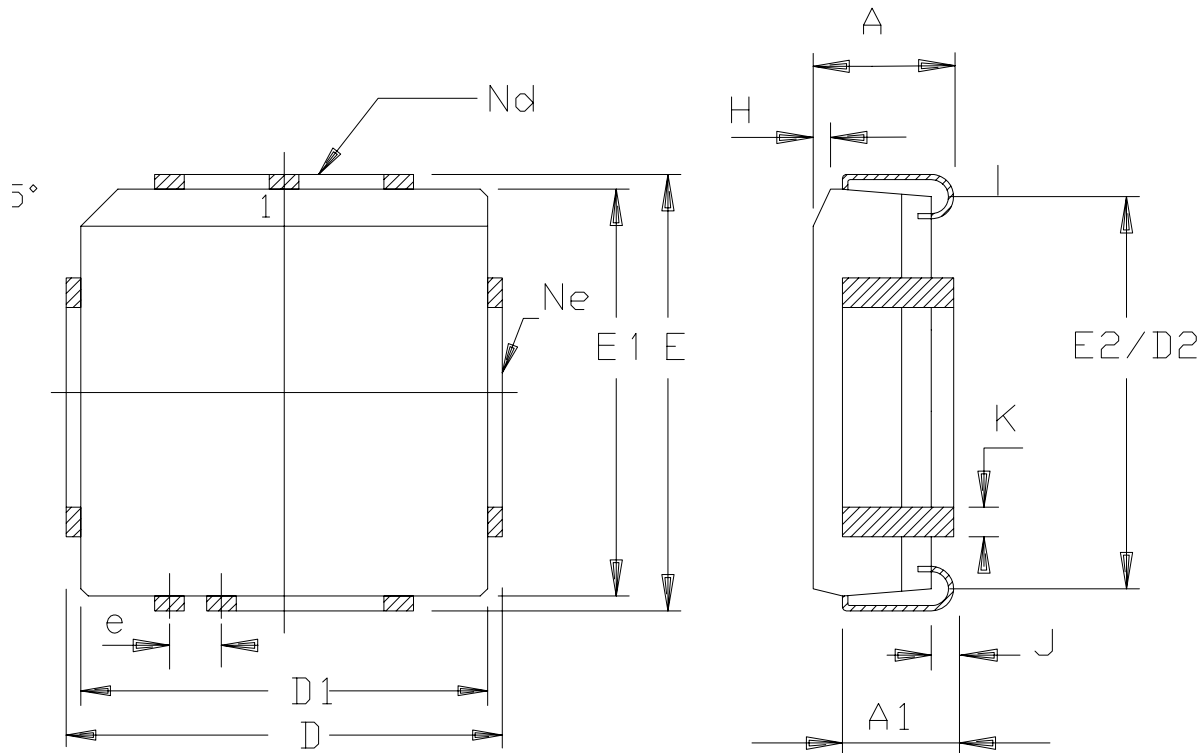
**4/ DATUM PLANE "H" LOCATED AT MOLD PARTING LINE AND
COINCIDENT WITH LEAD, WHERE LEAD EXITS PLASTIC BODY AT
BOTTOM OF PARTING LINE.**

5/ DATUM "A" AND "D" TO BE DETERMINED AT DATUM PLANE H.

**6/ DIMENSION " f " DOES NOT INCLUDE DAMBAR PROTUSION ALLOWABLE
DAMBAR PROTUSION SHALL BE 0.08mm/.003" TOTAL IN EXCESS OF THE
" f " DIMENSION AT MAXIMUM MATERIAL CONDITION .
DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT.**

PLCC28

28 PINS PLCC



| | MM | | INCH | |
|---------|--------|--------|-------|-------|
| A | 4. 20 | 4. 57 | . 165 | . 180 |
| A1 | 2. 29 | 3. 04 | . 090 | . 120 |
| D | 12. 32 | 12. 57 | . 485 | . 495 |
| D1 | 11. 43 | 11. 58 | . 450 | . 456 |
| D2 | 9. 91 | 10. 92 | . 390 | . 430 |
| E | 12. 32 | 12. 57 | . 485 | . 495 |
| E1 | 11. 43 | 11. 58 | . 450 | . 456 |
| E2 | 9. 91 | 10. 92 | . 390 | . 430 |
| e | 1. 27 | BSC | . 050 | BSC |
| H | 1. 07 | 1. 42 | . 042 | . 056 |
| J | 0. 51 | — | . 020 | — |
| K | 0. 33 | 0. 53 | . 013 | . 021 |
| Nd | 7 | | 7 | |
| Ne | 7 | | 7 | |
| PKG STD | | 00 | | |

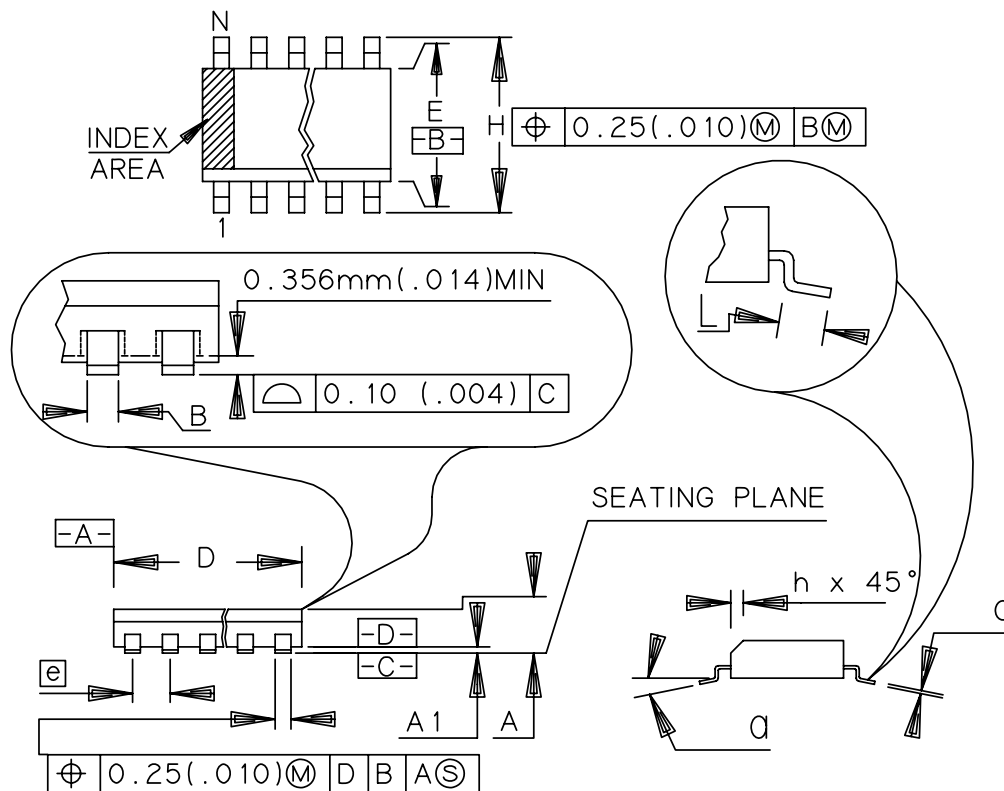
STANDARD NOTES FOR PLCC

1/ CONTROLLING DIMENSIONS : INCHES

2/ DIMENSIONING AND TOLERANCING PER ANSI Y 14.5M - 1982.

**3/ "D" AND "E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTUSIONS.
MOLD FLASH OR PROTUSIONS SHALL NOT EXCEED 0.20 mm (.008 INCH) PER
SIDE.**

SOIC24



| | MM | | INCH | |
|----|-------|-------|------|------|
| A | 2.35 | 2.65 | .093 | .104 |
| A1 | 0.10 | 0.30 | .004 | .012 |
| B | 0.35 | 0.49 | .014 | .019 |
| C | 0.23 | 0.32 | .009 | .013 |
| D | 15.20 | 15.60 | .599 | .614 |
| E | 7.40 | 7.60 | .291 | .299 |
| e | 1.27 | BSC | .050 | BSC |
| H | 10.00 | 10.65 | .394 | .419 |
| h | 0.25 | 0.75 | .010 | .029 |
| L | 0.40 | 1.27 | .016 | .050 |
| N | 24 | | 24 | |
| α | 0° | | 8° | |

07/27/07



Atmel Nantes S.A.
La Chantrerie - BP 70602
44306 Nantes Cedex 3 - France

TITLE
TD, 24 - Lead, 0.300" Body Width
Plastic Gull Wing Small Outline Package (SOIC)

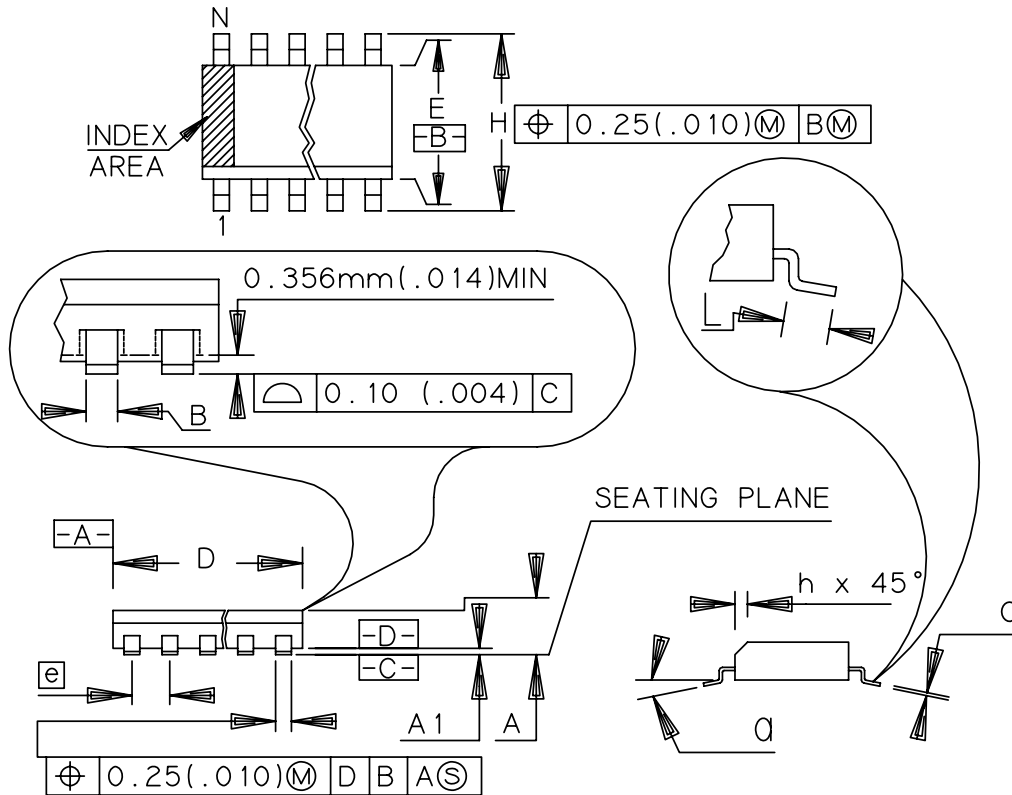
DRAWING No.

TD

REV.

A

SOIC28



| | MM | | INCH | |
|----|-------|-------|------|------|
| A | 2.35 | 2.65 | .093 | .104 |
| A1 | 0.10 | 0.30 | .004 | .012 |
| B | 0.35 | 0.49 | .014 | .019 |
| C | 0.23 | 0.32 | .009 | .013 |
| D | 17.70 | 18.10 | .697 | .713 |
| E | 7.40 | 7.60 | .291 | .299 |
| e | 1.27 | BSC | .050 | BSC |
| H | 10.00 | 10.65 | .394 | .419 |
| h | 0.25 | 0.75 | .010 | .029 |
| L | 0.40 | 1.27 | .016 | .050 |
| N | 28 | | 28 | |
| α | 0° | | 8° | |

07/27/07

NOTES: SOIC STANDARD NOTES

1. DIMENSIONING & TOLERANCING CONFORM TO ASME Y14.5M. – 1982.
2. "D" AND "E" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTUSIONS.
MOLD FLASH OR PROTUSIONS SHALL NOT EXCEED 0.15mm (0.006 INCH) PER SIDE.
3. THE CHAMFER "h" IS OPTIONAL.

Datasheet Revision History

Changes from 4126C-10/02 to 4126D - 04/03

1. Changed the endurance of Flash to 100, 000 Write/Erase cycles.
2. Added note on Flash retention formula for V_{IH1} , in Section "DC Parameters for Standard Voltage", page 141.Changes from 4129F-11/02 to 4129G-04/03
1. Changed the endurance of Flash to 100, 000 Write/Erase cycles.
2. Added note on Flash retention formula for V_{IH1} , in Section "DC Parameters for Standard Voltage", page 141.

Changes from 4126D - 05/03 to 4126E - 10/03

1. Updated "Electrical Characteristics" on page 140.
2. Corrected Figure 39 on page 82.

Changes from 4126E - 10/03 to 4126F - 12/03

1. Changed value of IPDMAX to 400, Section "Absolute Maximum Ratings", page 140.
2. PCA , CPS0, register correction, Section "PCA Registers", page 121.
3. Cross Memory section added. Section "Operation Cross Memory Access", page 44.

Changes from 4126F - 12/03 4126G - 08/04

1. Figure clock-out mode modified see, Figure 30 on page 65.
2. Corrected error in Table 51 on page 70, (1.25ms to 1.25s) for Time-out Computation.
3. Added explanation on the CAN protocol, see Section "CAN Controller", page 73.

Changes from 4126G - 08/04 to 4126H - 01/05

1. Various minor corrections throughout the document.

Changes from 4126H - 01/05 to 4126I 11/05

1. Added Green product ordering information.

Changes from 4126I to 4126J 05/06

1. Minor corrections throughout the document.

Changes from 4126J to 4126K 11/07

1. Updated Package drawings.

Changes from 4126K 11/07 to 4126L 02/08

1. Removed non-green part numbers from ordering information.

Table of Contents

| | |
|--|-----------|
| Features | 1 |
| <u>Description</u> | 2 |
| Block Diagram | 2 |
| Pin Configurations | 3 |
| Pin Description..... | 5 |
| I/O Configurations | 7 |
| Port Structure | 7 |
| Read-Modify-Write Instructions | 8 |
| Quasi Bi-directional Port Operation | 8 |
| SFR Mapping | 10 |
| Clock | 16 |
| Description | 16 |
| Register | 19 |
| Power Management | 20 |
| Reset Pin | 20 |
| At Power-up (cold reset)..... | 20 |
| During a Normal Operation (Warm Reset) | 21 |
| Watchdog Reset | 21 |
| Reset Recommendation to Prevent Flash Corruption | 22 |
| Idle Mode | 22 |
| Power-down Mode | 22 |
| Registers | 25 |
| Data Memory | 26 |
| Internal Space | 26 |
| Dual Data Pointer | 28 |
| Registers | 29 |
| EEPROM Data Memory | 31 |
| Write Data in the Column Latches..... | 31 |
| Programming | 31 |
| Read Data | 31 |
| Examples..... | 32 |
| Registers | 33 |

| | |
|---|-----------|
| Program/Code Memory | 34 |
| Flash Memory Architecture | 34 |
| Overview of FM0 Operations..... | 36 |
| Registers | 42 |
| Operation Cross Memory Access | 44 |
| Sharing Instructions..... | 45 |
| In-System Programming (ISP) | 47 |
| Flash Programming and Erasure | 47 |
| Boot Process | 48 |
| Application-Programming-Interface | 48 |
| XROW Bytes | 49 |
| Hardware Conditions | 49 |
| Hardware Security Byte..... | 50 |
| Serial I/O Port | 51 |
| Framing Error Detection | 51 |
| Automatic Address Recognition | 52 |
| Given Address..... | 52 |
| Broadcast Address | 53 |
| Registers | 54 |
| Timers/Counters | 57 |
| Timer/Counter Operations | 57 |
| Timer 0 | 57 |
| Timer 1 | 59 |
| Interrupt | 60 |
| Registers | 61 |
| Timer 2 | 64 |
| Auto-Reload Mode | 64 |
| Programmable Clock-Output..... | 65 |
| Registers | 66 |
| Watchdog Timer | 69 |
| Watchdog Programming..... | 70 |
| Watchdog Timer During Power-down Mode and Idle | 71 |
| Register | 71 |
| CAN Controller | 73 |

| | |
|---|-------------------|
| CAN Protocol..... | 73 |
| CAN Controller Description | 77 |
| CAN Controller Mailbox and Registers Organization | 78 |
| <i>CAN Controller Management</i> | <i>80</i> |
| IT CAN Management..... | 81 |
| Bit Timing and Baud Rate | 84 |
| Fault Confinement | 86 |
| Acceptance Filter..... | 87 |
| Data and Remote Frame | 88 |
| Time Trigger Communication (TTC) and Message Stamping | 89 |
| CAN Autobaud and Listening Mode | 90 |
| Routine Examples | 90 |
| CAN SFRs | 93 |
| Registers | 94 |
| <i>Programmable Counter Array (PCA)</i> | <i>114</i> |
| PCA Timer | 114 |
| PCA Modules | 116 |
| PCA Interrupt..... | 117 |
| PCA Capture Mode | 117 |
| 16-bit Software Timer Mode | 118 |
| High Speed Output Mode | 119 |
| Pulse Width Modulator Mode | 119 |
| PCA Registers..... | 121 |
| <i>Analog-to-Digital Converter (ADC)</i> | <i>126</i> |
| Features | 126 |
| ADC Port1 I/O Functions | 126 |
| VAREF | 126 |
| ADC Converter Operation | 127 |
| Voltage Conversion | 128 |
| Clock Selection..... | 128 |
| ADC Standby Mode..... | 128 |
| IT ADC Management..... | 129 |
| Routine Examples | 129 |
| Registers | 130 |
| <i>Interrupt System</i> | <i>132</i> |
| Introduction..... | 132 |
| Registers | 134 |
| <i>Electrical Characteristics</i> | <i>140</i> |

| | |
|---|------------|
| Absolute Maximum Ratings..... | 140 |
| DC Parameters for Standard Voltage..... | 140 |
| DC Parameters for A/D Converter..... | 142 |
| AC Parameters..... | 143 |
| Ordering Information | 146 |
| Package Drawings | 147 |
| VQFP32..... | 147 |
| PLCC28..... | 149 |
| | 150 |
| SOIC24..... | 151 |
| SOIC28..... | 152 |
| Datasheet Revision History | 154 |
| Changes from 4126C-10/02 to 4126D - 04/03 | 154 |
| Changes from 4126D -05/03 to 4126E - 10/03 | 154 |
| Changes from 4126E - 10/03 to 4126F - 12/03..... | 154 |
| Changes from 4126F - 12/03 4126G - 08/04 | 154 |
| Changes from 4126G - 08/04 to 4126H - 01/05 | 154 |
| Changes from 4126H - 01/05 to 4126I 11/05 | 154 |
| Changes from 4126I to 4126J 05/06 | 154 |
| Changes from 4126J to 4126K 11/07..... | 154 |
| Changes from 4126K 11/07 to 4126L 02/08 | 154 |
| Table of Contents | i |



Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe

Atmel Sarl
Route des Arsenalux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia

Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan

9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantreterie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards

Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive

Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/ High Speed Converters/RF Datacom

Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests

www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© Atmel Corporation 2008. All rights reserved. Atmel®, logo and combinations thereof, and Everywhere You Are® are the trademarks or registered trademarks, of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.



Printed on recycled paper.

Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.

С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические), за счёт заключения дистрибьюторских договоров

Мы предлагаем:

- Конкурентоспособные цены и скидки постоянным клиентам.
- Специальные условия для постоянных клиентов.
- Подбор аналогов.
- Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.
- Приемлемые сроки поставки, возможна ускоренная поставка.
- Доставку товара в любую точку России и стран СНГ.
- Комплексную поставку.
- Работу по проектам и поставку образцов.
- Формирование склада под заказчика.
- Сертификаты соответствия на поставляемую продукцию (по желанию клиента).
- Тестирование поставляемой продукции.
- Поставку компонентов, требующих военную и космическую приемку.
- Входной контроль качества.
- Наличие сертификата ISO.

В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.

Конструкторский отдел помогает осуществить:

- Регистрацию проекта у производителя компонентов.
- Техническую поддержку проекта.
- Защиту от снятия компонента с производства.
- Оценку стоимости проекта по компонентам.
- Изготовление тестовой платы монтаж и пусконаладочные работы.



Тел: +7 (812) 336 43 04 (многоканальный)

Email: org@lifeelectronics.ru

www.lifeelectronics.ru